APPLE BASIC
應用設計



張其邦 譯

博文出版社出版

APPLE BASIC
應用設計

張其邦 譯

博 文 出 版 社 出 版

APPLE BASIC 應用設計

編譯者：張　其　邦
出版者：博　文　出　版　社
發行者：博　文　出　版　社
九龍大連排道872號
印刷者：鴻　文　印　刷　廠
香港柴灣工廠大厦五樓

定價：H. K. $ 35.00

序

致讀者：

在市面上有許多討論Apple][的書，並且隨著您購買的機器也有幾本Apple 機器提供的手册，它們的內容大都是相關於圖形及指令的解說，但是卻沒有一本書教您如何設計程式。

我們希望能夠提供給您設計程式的方法及技巧；許多講述設計程式的書都只是將指令集告訴您，當然我們也免不了這麼做，但是我們將專注於教導您如何使用這些指令配合著您的機器設計出有效率的程式。

在我們對微電腦的世界做一個簡短的介紹之後，我們就要打穩您設計程式的基礎了；像是計算的過程，資料的型式、變數、輸出輸入等等都是您在設計程式之前必須瞭解的。第三章講的是程式的發展方式；第四章則討論程式的結構，同時對您講解一些實際的例子。

只要在您將程式設計完成之後，您就必須測試程式中的錯誤；因此，您接下來要學習的就是測試及處理錯誤的方法，這些工作使您的程式完整並且有效。

這些觀念我們將在第十二章做一個總結，我們將使用一個實際的商業上使用的程式告訴您發展程式的方法及途逕。

在您研究完了這本書之後，相信您已經可以設計一個完整並有效的程式了；這時候，您就可以將本書視爲一本參考手册，供您設計程式時參考之用。同時，我們在書中也提供您一些有用的副程式，使您在設計程式時能夠省力很多。

在本書中，我們設計程式所擁有的硬體設備及軟體如下：

* －Apple][電腦(48K)
* －Disk][磁碟機(DOS 3.3)
* －APPLESOFT][BASIC

*一整數 BASIC

*一 APPLEWRITER

所有在書中的例子都是使用 APPLESOFT][BASIC 語言設計的，如果有其他的情形，我們將做說明。

並不是書中所有的章節您都必須研讀完畢後才能設計程式，有關於硬體方面的章節只是希望能夠使您更加的深入瞭解罷了！

祝您

設計程式愉快

•張其邦•

目　錄

第1章　介　紹

1.0　大　綱 …… 1
1.1　歷　史 …… 1
1.2　APPLE 電腦公司 …… 4
1.3　目前的趨勢 …… 5
1.4　總　結 …… 5

第2章　基　礎

2.0　大　綱 …… 7
2.1　電　腦 …… 7
2.2　計算的過程 …… 9
2.2.1　範例程式 …… 11
2.3　立即式及間接式（ Immediate and Deferred modes ）…… 14
2.4　BASIC 程式語言 …… 15
2.5　資料的型式 …… 15
2.5.1　數　字 …… 16
2.5.2　字　串 …… 16
2.5.3　變　數 …… 17
2.6　變數值的設定 …… 19
2.7　流程的控制 …… 20
2.7.1　由下往上（ Bottom-up）的方式 …… 21
2.7.2　由上往下（ Top-Down ）的方式 …… 21
2.8　輸　入 …… 22
2.8.1　INPUT 敍述 …… 22

2.8.2 GET 敍述…… 23
2.8.3 DATA 敍述…… 23
2.8.4 顯示訊息…… 24
2.8.5 螢幕顯示…… 24
2.9 輸 出…… 26
2.9.1 PRINT 敍述…… 26
2.9.2 設定格式…… 26
2.9.3 螢幕顯示的安排…… 27
2.10 處理過程…… 28
2.10.1 程 序…… 28
2.10.2 修正錯誤…… 29
2.11 設計程式的一般規則…… 30
2.12 總 結…… 31

第3章 程式的發展

3.0 大 綱…… 32
3.1 組織方式的考慮…… 32
3.1.1 速 度…… 33
3.1.2 整合性…… 34
3.1.3 主程式…… 34
3.1.4 副程式…… 35
3.2 程式的結構…… 36
3.2.1 順 序…… 39
3.2.2 FOR-NEXT 迴圈…… 40
3.2.3 CASE的結構…… 45
3.2.4 IF-THEN-ELSE…… 46
3.2.5 不停止的迴圈…… 47
3.3 輸入/輸出的格式…… 51
3.3.1 控制印出的位置…… 52
3.3.2 格式的控制…… 53

3.3.3　大/小寫字體 …… 54
3.4　外圍設備的考慮 …… 55
3.4.1　APPLEⅡ的BUS …… 55
3.4.2　串列介面卡(Serial interfaces) …… 55
3.4.3　平行介面卡(Parallel Interface) …… 60
3.4.4　訊號交換介面卡(Communications Interface) 61
3.5　總　結 …… 62

第4章　程式的結構

4.0　大　綱 …… 63
4.1　記憶體的使用 …… 64
4.2　變數表 …… 69
4.3　字串表 …… 70
4.4　程式的編譯 …… 71
4.5　手　册(MENU) …… 74
4.6　副程式 …… 75
4.6.1　一行副程式 …… 75
4.6.2　中型副程式 …… 75
4.6.3　大型副程式 …… 76
4.7　程式的架構 …… 77
4.8　延　伸 …… 80
4.9　總　結 …… 80

第5章　程式的測試

5.0　大　綱 …… 81
5.1　主要的程式段 …… 82
5.2　副程式段 …… 82
5.3　程式系統 …… 83
5.4　處理錯誤的程式段 …… 83
5.4.1　輸入的錯誤 …… 83

5.4.2 處理時的錯誤……86
5.4.3 輸出的錯誤……86
5.4.4 磁碟的錯誤……86
5.5 程式設計師的測試工作……86
5.6 Alpha / Beta 測試觀點……87
5.7 總　結……88

第6章　錯誤的處理

6.0 大　綱……89
6.1 一個乾淨的機器……89
6.2 範圍的限定……91
6.3 ONERR 敍述……93
6.4 輸入的限定……95
6.5 總　結……95

第7章　應用程式

7.0 大　綱……96
7.1 排列的程式……96
7.1.1 Bubble 的排列方式……97
7.1.2 插入的排列方式……100
7.2 搜尋資料……103
7.2.1 線性搜尋……103
7.2.2 二分法的搜尋……105
7.2.3 字串的搜尋……109
7.3 技術與格式……111
7.4 各式各樣的程式……112
7.4.1 分解一個數字……112
7.4.2 Horners 法則……114
7.4.3 求極大及極小值……116
7.4.4 Soundex方法……117

7.4.5 Print Using……118
7.4.6 過濾數字的副程式……121
7.4.7 過濾字串的副程式……122
7.5 總 結……123

第8章 作業系統

8.0 大 綱……124
8.1 磁帶系統（Cassette System）……127
8.2 磁帶系統……128
8.2.1 一般地考慮……128
8.2.1.1 設定磁片的格式……129
8.2.1.2 SAVE及LOAD程式……130
8.2.1.3 RENAME及DELETE命令……132
8.2.1.4 CATALOG命令……133
8.2.1.5 LOCK及UNLOCK命令……134
8.2.1.6 FP及INT指令……134
8.2.1.7 磁碟的格式……135
8.2.2 檔的結構……138
8.2.2.1 順序處理檔……139
8.2.2.2 隨機處理檔……149
8.2.2.3 二進位檔……163
8.2.2.4 EXEC檔……164
8.3 總 論……167

第9章 圖 形

9.0 大 綱……169
9.1 低解析度圖形……169
9.2 高解析度圖形……176
9.3 造型表（Shape Tables）……181
9.4 將圖形印製在紙上……184

9.4.1 非撞擊式的列表機……186
9.4.2 撞擊式的列表機……195
9.4.2.1 點矩陣式的列表機……195
9.4.2.2 字型的列表機……196
9.5 總 結……199

第10章 將程式做好完整的記錄

10.0 大 綱……200
10.1 分析式的記錄工作……201
10.2 系統的記錄工作……202
10.3 程式的記錄工作……202
10.4 幫助操作及使用者的記錄工作……203
10.5 流程圖……203
10.6 總 結……204

第11章 各種程式

11.0 大 綱……205
11.1 數學的程式……205
11.1.1 調換行與列的位置……206
11.1.2 矩陣的乘法……208
11.1.3 NEWTON-RAPHSON 方法……210
11.1.4 搜尋方法（SEARCH METHOD）……211
11.1.5 BISECTION 方法……212
11.1.6 FIBONACCI 數列……214
11.1.7 高斯消去法（Gauss Elimination）……216
11.2 轉換的程式……220
11.2.1 溫度……220
11.2.2 度度量轉換成弳度量（ANGLES TO RADIANS）……221

11.2.3 弳度量轉成度度量（RANDIANS TO ANGLES）……222
11.2.4 公 制……223
11.3 遊戲程式……231
11.3.1 APPLE ALLEY 遊戲……232
11.3.2 NIM GAME 遊戲……238
11.3.3 QUBE FOUR 遊戲……241
11.3.4 REVERSE 程式……249
11.3.5 生命週期（BIORHYTHM）……251
11.3.6 日 曆（CALENDAR）……257
11.3.7 唱歌的程式……259
11.4 總 結……263

第12章 滙 總

12.0 大 綱……264
12.1 選擇問題……264
12.2 對問題的描述……265
12.3 設計解決的程式……273
12.4 程 式……275
12.5 確定程式的正確性……329
12.6 程式系統的記錄手册……330
12.6.0 介紹的部份……330
12.6.1 第一章……333
1. 每期應還金額（REGULAR PAYMENT ON NOTE）……334
2. 帳上的現值（PRESENT VALUE ON NOTE）……335
3. 定期付款的次數（# OF PAYMENTS）……336
4. 年利率（ANNUAL INTEREST RATE）……336
12.6.2 第二章……337

1. 期末本利總額（ FUTURE VALUE ON SAVINGS ）…………338
2. 存款現值（ PRESENT VALUE ON SAVINGS ）…………338
3. 存款的時間數（ # OF PAYMENTS ）…………339
4. 年利率（ ANNUAL INTEREST RATE ）…………340
12.6.3 第三章…………341
1. 年金終值（ FUTURE VALUE OF ANNUITY）…………341
2. 每期固定付款數額（ PAYMENT EACH PERIOD ）…………343
3. 付款的次數（ # OF PAYMENTS ）…………344
4. 年利率（ ANNUAL INTEREST RATE）…………345
12.6.4 第四章…………346
12.6.5 第五章…………347
12.6.6 第六章…………348
12.6.7 第七章…………350
12.6.8 第八章…………351
12.6.9 第九章…………353
12.6.10 第十章…………355
12.6.11 附　錄…………357
附錄A　列表機的問題…………357
附錄B　理論及數學公式…………358
附錄C　範例報表…………364
12.7 總　結…………371
附錄A　APPLESOFTⅡ 命令…………372
附錄B　整數BASIC的命令…………394
附錄C　POKE．PEEK 及 CALL…………406
附錄D　磁碟機作業系統的命令…………420
附錄E　錯誤的訊息…………435

附錄F　保留字………………………………………………443
附錄G　編輯命令……………………………………………445
附錄H　設定的函數…………………………………………477

第 1 章

介　紹

1.0 大　綱

在這一章中，我們將專注於微電腦歷史的介紹，尤其是 Apple 電腦公司的描述；如果您對這些故事都已經知道了，那麼您可以跳過這一章。

1.1 歷　史

如果在 20 年前有一個先知告訴世人說：“我將在一個店中只花費美金 2.5 元就可以買到一個微處理機”。這一句話將使得人們認爲他是一個傻瓜；但是，事實上在 1979 年的春天這個神話已經在 Texas A&M University 實現了。

微處理機的實際誕生日期大約是在 1966 年的五月 5 日；到了 1968 年的四月，Jim Sutherland 發表了一個新的機器，這個機器有四個暫存器（ registers ），每個字（ words ）有 4 個數元（ bit ）；並且有 8 K bytes 的記憶體，至於可以使用的指令共有 18 個，它的速度是 180KHZ。也就是在 1968 年，許多人從事個人電腦（ personal computers ）的設計工作，而這些個人電腦的一般特性就是：

* ＊－500KHZ 的速度
* ＊－4 ～ 8 K 的記憶體
* ＊－3 ～ 4 個暫存器

＊一 I/O 的型式與打字機相似

Intel公司對於微處理機的發展可以說是貢獻最大的了。在1969 年，有一個名叫 Busicom 的日本公司與 Intel 公司簽了發展單一晶片型式的計算機的合約，這個晶片內由具有4 個數元的字及4 個 IC 所組成，這些 IC 分別是處理機（ processor ），僅讀記憶體（ ROM ）上面連接著 I/O ，隨機處理記憶體（ RAM ）上面接著 I/O ，以及一個移動的暫存器（ shift register ）。這個處理機就被稱爲4004 ，並且以美金200 元的代價出售，而這也就是微電腦的開始了。

幾乎在同時， Datapoint 這個製造終端機（ terminals ）的公司也與 Intel 及 Texas Instruments 公司簽訂發展單一晶片處理機的合約；Intel 公司成功地完成了這個工作，但是它對於 Datapoint 公司而言速度似乎太慢了，這使得 Intel 失去了這個顧客；但是在 1971 年， Intel 決定將這個晶片編號爲8008 並且仍然以200 元的代價出售；這就是第一代積體電路的微處理機。

8008 使用的字是8 個數元，並且具有較4004更有效的指令集（ ins - truction set），它較4004 更具有一般性（也就是說許多的邏輯動作都由處理機完成了），8008 的指令集與 minicomputer 的相似，包含有打斷的結構（ interrupt ）以及可以直接地指到16K 的記憶體位置（ address ）。

第一個電腦的套件由at National Radio Institutes的Louis E. Frenzel推出，這個機器具有52 個(TTL)的 IC ，並且具有32 乘8 的 IC 記憶體，15 個指令，並且具有一個操作員的面板。這套套件售價是503 美元。

1972 年是一個眞正的轉捩點；在這一年中，Don Tarbell 在他的系統中加入了編輯（ Editor ）及組合程式（ Assembler ），這麼做的重要性就在於它使與電腦的聯繫工作變得簡易的多了。這時候，市場上有 Intel 的8008 ，並且 TTL 的價格也大幅度的跌落；因此，一切的準備工作都已就緒了。

同時，在這一年中許多積體電路的製造商也都開始瞭解到了這個新市場

並且開始發展它們的處理機。

在1973年，Intel 公司推出了8080的微處理機，這個微處理機後來變成最受歡迎並且效用最廣的微處理機。8080具有40 pin 的 DIP，並且有一個眞正雙向的資料傳送器（ data bus ）以及一套可擴增的指令集。

在8080 推出之後不久，Motorola 公司推出了它們的產品－6800，6800設計的方式與8080完全不同，它只需要一個電源供應器（ power supply ）－這成為一種標準的特性，需要的控制邏輯綫路較少，並且具有與大型 minicomputers 相似的指令集。這個階段您可以視它為微電腦的第二代。

在1975及1976年，我們看到微處理機的迅速發展，在這段時間內所生產的微處理機有 Zilog 的 Z80，MOS Technology 的6502，TI 的9900，以及8080及6800改進後的產品，這一段時間就被稱為第三代。

在1977年，我們看到第一套微電腦的系統；而1977及1978年則是微處理機及微電腦快速進步的時代，並且不單單是處理機的進步，就連記憶體等晶片都有了很大的進步。

在1979 年我們見到微電腦及微處理機的第四代，它們分別是 Zilog 的8000系列，Motorola 的68000，以及 Intel 的8086及8088；同時也有了彩色顯示器及微型磁碟機（ Floppy Disk ）的控制卡等週邊設備的產生。

由於有了以上的產品，使得我們在1980 年看到了多人使用（ multi-user ）及多重作業（ multi-tasking ）的系統，它們都是配合著硬式磁碟機（ hard disks ）而產生的。

接著我們就見到了具有更大功能的機器了，像 APPLE III 就具有了可擴增的記憶體及彩色的顯示幕，更加上精細的作業系統（ operating system ）－ SOS。同時，TANDY 企業的 TRS 80 MODEL II 的機器具有更大的記憶體及螢幕顯示的能力。

許多只具有一塊板的電腦同時也被介紹給大衆；由於磁碟的迅速進步，使得我們能夠處理的事務更加的精細；例如 Winchester 公司5″的硬式磁碟機就可以容納4.53 Mb 的資料。

許多的日本機器同時也施予美國機器很大的壓力；例如，NEC，Sony

及 Mitsubishi 等廠家都推出了它們獨自的機器，也具有多方面的功能。

1.2 APPLE電腦公司

在加州的 Homebrew Computer Club，可以說是微電腦工業的推動者，在他們的會員之中具有許多位電腦的專才。 Apple 公司的創立者Jobs 先生及 Wozniak 先生他們也是屬於這個俱樂部，他們在設計完了 Apple 電腦之後，就成立了 Apple 電腦公司，該公司名稱的由來是由於 Jobs 先生有一個水果店而他們希望取的名字是具有非技術性及友善的，故而他們取了 Apple 這個名字。

差不多在同一個時間誕生了兩個雜誌，第一個就是 BYTE（ 1975 年的九月），另外一個則是 Southern California Computer Club 雜誌（也就是現在的 Interface Age ），這些雜誌使這項工業逐漸被世人所瞭解。

在 1975 年的夏天， MOS 技術公司將6502晶片在市面上發售，而這正是 Apple 電腦的 CPU 。在1975 年的11月 Apple I 終於與世人見面了，它所具有的只是一個鍵盤及顯示幕。

在 1976 年的春天，16 K的 RAM 晶片應市，同時 Apple I 也採用了這項產品，而這就是目前我們所見到的 Apple][的雛型。

在 1977 年的4 月5 日， Apple][正式推介給大衆，同年的六月第一部的 Apple][被賣出。

現在是我們介紹 Apple 電腦公司組成人員的時候了。

* — Steve Jobs 先生：原先在 ATARI 公司工作，也是 "Breakout" 遊戲的發明人。
* — Steve Wozniak 先生：原先在 HP 公司的 Advanced Calculator Group 工作，也是 Apple 電腦的設計者。
* — Mike Scott 先生：原先在National Semiconductor公司任 Hybrids and Transducers 的主任。

*— Mike Markulla 先生：是原先 Intel 公司的退休市場主管。

*— Rod Holte 先生：原先在 ATARI 公司任工程師主管。

在 1978 年的六月，我們見到 Apple 電腦公司銷售的月份累積達到一百萬美元；到目前爲止，這個公司已經成爲電腦工業的一項傳奇了。

1.3 目前的趨勢

在過去的 25 年間，我們發現電腦的體積及價格逐漸降低，而所能擔負的工作能力卻有了相當大的進步。

那麼在未來我們希望能達到什麼樣的境界呢？這個問題如果專注於某一方面來講，似乎較難回答，但是就一般情形來說，我們將看到同樣大小的電腦具有更多方面的功能，而證據就是 Zilog 公司的 Z8000及 Motorola 公司的 68000 微處理機的推出。

同樣地，微處理機已經在家庭用品中廣泛地被應用了；例如洗衣機、乾衣機、電視等等；因此，在未來我們將可見到電腦化的家庭。

現在我們見到日漸增多的軟體系統被“燒”在硬體內，這使得成本及機器的大小得以降低。

未來的商業機構將變成資訊交換的中心，它們將使用電腦做爲與賣方、買方、製造商間的聯繫及儲存資料的工具，而它們之間互相使用電話綫做連接就可以了。

Apple 電腦公司在 1980 年推出 APPLE ///，而即將推出的 APPLE IV則是使用 Motorola 68000 的微處理機。

1.4 總 結

在本章中我們大略的描述了微電腦工業及 Apple 電腦公司的歷史。

我們也見到了在很短的時間內晶片進步的情形。

這一章的目的就是希望能夠使您有從何處開始及未來將如何發展的認識。

第2章 基礎

2.0 大 綱

這一章講述的是基本的電腦型式，替您打好設計程式的基礎，使您能夠將訊息顯示在螢光幕上。我們教您設計 Apple][程式語言的方法並不是要求您將所有的指令都背起來，而是要使您能夠靈活運用它們（ BASIC ），而這種方法將使您在轉換到其他語言時能較爲簡易。

2.1 電 腦

在這一節中，我們將討論微電腦的結構，討論的方向將由一般的結構到一些常被使用的晶片的結構。

一個電腦的基本結構就如圖２－１所顯示的一般，在圖中我們看到五個主要的部份（有些人視其爲四個部份）。它們分別是：

* 一控制部門（ Control Unit ）
* 一算術及邏輯部門（ Arithmetic and Logic Unit）
* 一記憶體（ Memory ）
* 一輸入部門（ Input ）
* 一輸出部門（ Output ）

控制部門控制著整個電腦系統。

算術及邏輯部門執行所有資料的運算工作。

記憶體做爲程式及資料的儲存單位。

輸入部門與外在的設備連接而將資料及程式輸入電腦內。

輸出部門則使電腦能夠將資料輸送到外在的設備。

控制部門與算術及邏輯部門聯合起來就成爲中央處理部門(Central Processing Unit)— CPU。

```
                  ***********
                  *         *
                  * Memory  *
                  *         *
                  ***********
                     !   !
                     !   !
*********         ***********         **********
*       *         *         *         *        *
* Input *--->*  Control *--->* Output *
*       *         *         *         *        *
*********         ***********         **********
                     !   !
                     !   !
                  ***********
                  *         *
                  *   ALU   *
                  *         *
                  ***********
```

圖2-1 典型的電腦系統

由結構的外觀來看，這些部門都是由一些 PC 板連接著許多的綫，然後包裝在一個盒子內而組成了電腦的硬體部份。

其他的盒子(例如 I/O 部份)再與電腦的主要部份連接起來形成整個電腦的完整組合，這種型式的結構一直延續到大約五年之前，才被 LSI (Large Scale Integration)的發展成功而取代。由於這種電子技術的進步使得先前許多盒子的組合只需要佔用¼平方英吋的一個晶片就可以做同樣的功能了，而這個晶片就被稱爲微處理機。

接下來我們要定義微處理機及微電腦的意義。在微電腦與微處理機間最大的分別是非常明顯的；電腦的意義就是代表微處理機及其他像是記憶體、輸入輸出設備等等的完整系統；因此，微處理機就是微電腦系統中的一個重要元件。

目前市面上有許多的廠家製造出售各式的微處理機，我們在表２－１中列出了主要的幾項微處理機及它們的製造商，如果您對這些沒有興趣，您可以不去看它，這對您接下去的學習工作沒有任何的影響。

	微處理機	製造商
1 chip	3870,3872	MOSTEK,Motorola,Fairchild
	8039,8048,8049,8748	Intel,RCA,NEC,AMD,Signetics
	TMS1000/1000C/1200	TI,Motorola
NMOS	8080,8080A,8085	Intel,TI,NEC,AMD,National
	6502	MOS Tech,Rockwell,Synertek
	6800,6800A,6801, 6802,6803,6809	Motorola,AMI,Fairchild, Fujitsu
	Z-80,Z-80A,Z-8	Zilog,MOSTEK,NEC
	2650	Signetics
CMOS	1802	RCA,Hughes
	6100	Intersil,Harris
16-bit plus chips	Z8000	Zilog,AMD
	TMS9900,9940	TI,AMI
	8086,8088	Intel
	LSI-11	DEC
	68000	Motorola

表2-1 微處理機的製造商

2.2 計算的過程

這一節討論的是電腦解決問題或是幫助您尋求答案的過程。對於程式設

計師而言，這個過程包含了下列的工作：

1. 選擇適合於電腦系統本身能力的問題。
2. 確實地描述問題的需要。
3. 設計解決問題的方法。
4. 用適當的程式語言描述解決的方法。
5. 將您的程式完整的記錄並說明清楚。

在微電腦上選擇適當的問題供其解決是一件與解決問題同樣困難的事，因爲您所選擇的問題必須是適合於這個電腦所具有的能力的，而您必須先對這個機器有相當深入地瞭解才可以。然而，下面是我們選擇問題時可供參考的兩個事實。

首先，問題是頗複雜的，當使用人力計算時，必須耗費許多的時間與精力，而且計算的過程往往是重複許多次的。如果問題本身非常的簡單而且並沒有許多的計算工作，那麼您花費在程式設計上面的精力往往是多餘的。需要注意的是，我們在本書中所討論的例子都是使用稍嫌瑣碎的程式解決的，這是爲了能使您逐漸的發展出您設計程式的能力。

其次，電腦只能協助解決具有詳細定義的問題，如果一個沒有仔細定義的問題交給電腦來解決，那麼電腦將無法給予您一個確實的答案。

典型的例子就是西洋棋的遊戲，電腦被要求與人下棋；經過許多年不斷的改良後，由於定義更加明確，而程式設計師也變得更加精進，使得這個電腦的軟體就如同人一般地聰明了。

我們所謂明確的定義問題，指的是所有的輸入、輸出，檔與檔的結構與聯繫，資料間互相的關連，以及錯誤的處理、螢幕的顯示格式等等都必須有詳細完整的定義才可以，否則程式設計後將無法如您所預期的得到結果。

現在相信已經是使用一個範例程式做解說的時候了，我們可以使用任何的程式語言設計這個程式，但是基於本書的目的，我們使用的是 Apple 的 BASIC 語言。

在您將程式設計完畢之後，您就必須做確定解答是否正確的工作，這並不只是除去語法的錯誤就可以了，您還必須檢查程式的邏輯、輸入的範圍

，以及測試答案是否正確等等，這個檢查的工作往往是費時且費力的。

最後，您必須做的工作就是將您的程式及所得到的結果做一份完整的記錄；在這個工作中，您將所有對程式的描述、分析、程式列以及一份範例的結果包含在這份記錄內，並且對於執行程式的方法也需要一份說明，這樣才能使往後的使用者在使用您的程式時充滿了信心。

如果您在設計程式時依照上面的方式，那麼您將發現它將使您省去許多不必要的時間浪費，並且使您的工作能夠順利的完成。

2.2.1 範例程式

我們現在要用一個簡單的程式作為範例，目前您也許還無法詳細地瞭解整個的程式，但是您將得到一個大致的概念。

A—問題

您想要使用 APPLE][找到一組五個正數中最大的數字。

B—步驟

* —找出五個正數。
* —由這一組數字中找出最大的數字。
* —將這個數字顯示在 CRT 上。
* —設定 CRT 的顯示格式；
    * —清除螢幕
    * —印出最大的數字
* —結束程式的執行。
* —檢查錯誤。
* —將解決的方法做成記錄。

C—解決方法的描述

將五個數字讀入記憶體內，將某一個變數設為負值，並將所有的數字與這個假設為極大值的負值相比較，並在需要的時候做重設極大值的工作。

將最後的極大數字顯示在螢光幕上，並且將顯示的格式定好。

D—程式

```
1 REM ***** FIND MAXIMUM ******
2 REM *
3 REM * WRITTEN BY:JL CAMPBELL
4 REM *       DATED:05/10/81
5 REM *
6 REM *    COMPUTER:APPLE II
7 REM *    LANGUAGE:APPLESOFT II
8 REM *
9 REM ***************************
10 TEXT: CLEAR: POKE 216,0: A=FRE(0)
20 DIM NR(5)
30 FOR I=1 TO 5: READ NR(I): NEXT I
32 DATA 5,9,11,4,3
40 MX= -1
50 FOR I=1 TO 5
60 IF MX > NR(I) THEN 80
70 MX = NR(I)
80 NEXT I
100 HOME: PRINT: PRINT TAB(16) "MAXIMUM":PRINT
110 PRINT "MAXIMUM NUMBER = ";MX
120 END
```

圖2-2 找極大值的程式

現在您就可以輸入這個程式並執行它了；檢查它的錯誤並將這個問題做成記錄；您應該注意的是，上面這個程式並非解決這個問題的唯一方法。

解決任何問題的第一步必定是對這個問題的分析。這個程式是由四個部份組成的；第一部份只是導引進入程式的工作——也就是行號 1 到 9 的部份，接著在行號 10 的地方將螢光幕清除乾淨，至於行號 20 則先設定五個存數字的記憶位置。

第二部份則將這些數字輸入並設立其他的變數值，這就是在行號30到40的地方。

第三部份就是程式所有過程中最重要的一部份，它執行程式重要的動作，由行號 50 開始到 80 結束。

最後則是第四部份，它將得到的結果顯示在螢光幕上，包含了行號 100 到 110 的程式行；至於 120 這一行，就是結束程式執行的程式行。

2.3 立即式及間接式（Immediate and Deferred modes）

Apple 有兩種不同的作業方式，它們分別是立即式的執行及間接式的執行。在立即式的情況下，任何合法的命令都被立即地執行；至於間接式的情況則是在執行某一個命令之後才將合法的命令執行，而這就是您在使用高階語言（如 BASIC ）執行程式的情形。當電腦的開關被打開後，您就可以執行立即式的命令了。

下面就是這兩種作業方式的例子。

```
PRINT "MY COMPUTER LIKES ME!"
MY COMPUTER LIKES ME!

PRINT "NOW IS THE TIME."
NOW IS THE TIME.
```

由於這兩個敍述都沒有行號，因此只要您按下了 RETURN 鍵，它們就即刻被執行了。 PRINT 這個命令就是命令電腦將包含在引號之內的資料印出來。

您也可以在立即式的情況下將 APPLE 視爲一個計算器，它將在您按下 RETURN 之後即刻地顯示出計算的結果。

加 法	PRINT 15+22 37
減 法	PRINT 450-225 225
乘 法	PRINT 50*12 600
除 法	PRINT 66/11 6
指數計算	PRINT 2^10 1024
混合計算	Print 4*3+10-5 17

答案將顯示在螢光幕的最左端，同時在您做算術時，您不必使用引號。

立即式的執行命令	間接式的執行命令
AUTO	END
CLR	FOR
CON	GOSUB
DEL	INPUT
HIMEM:	NEXT
LOAD	RETURN
LOMEM:	
MAN	
NEW	
RUN	
SAVE	

表2-2 整數Basic命令的限制

間接式的執行則要求在所有的敍述都被輸入後，才做執行的動作，而這正就是程式發展的方式。

```
100 PRINT "MY COMPUTER LIKES ME!"
110 END
```

這個例子就是一個小的 BASIC 程式，當您打入了 RUN 命令後，它就會將引號之間的句子顯示在螢光幕上。在您將程式輸入記憶體內之後，您由鍵盤打入 RUN 並按下 return 鍵，那麼就會使這個程式被執行了。

立即式的執行命令	間接式的執行命令
DATA	注意：所有其他的敍述及命令 statements and commands
DEF FN	
GET	
INPUT	
ONERR GOTO	
RESUME	

表2-3 Applesoft II Basic命令的限制

2.4 BASIC程式語言

BASIC 是一種在電腦上使用的高階語言，它的全名就是 Beginner's All-purpose Symbolic Instruction Code(BASIC)，這種語言是於 1967 年由John Kemeny及 Thomas Kurtz 在 Dartmouth College 發展成功的。從這以後，這種語言就成爲微電腦的標準語言了。

在 Apple][上有兩種型式的 BASIC 語言，它們分別是整數 BASIC 及 APPLESOFT][，每一種的 Apple][電腦都會具有這兩種語言或是其中之一；因此，這使得 Apple][能夠具有多重語言的功能；事實上，Apple][上能夠應用的語言尙有 PASCAL，PILOT,FORTRAN 等等其他的高階語言，只是在本書中我們並不討論它們就是了。

整數 BASIC 語言的標示字(prompt character)是“ > ”，而 APPLESOFT][則是“] ”，因此您在開機時就可以分辨出來您的機器擁有的是那一種語言了。

整數 BASIC 只能從事整數的運算，這表示當您在做算術時計算⅔，您得到的結果會是1 ；在 4.5 乘上 3 時 (405＊3)，您將不被允許做這個計算，因爲 4.5 是一個實數而非整數。

2.5 資料的型式

我們在這一節中討論資料的型式以及它們在程式中的用途。在 Apple Ⅱ中，我們可以使用三種不同型式的資料，它們分別是實數 (Real)，整數 (Integer) 以及字串 (String) 。

對於程式設計師來說，瞭解資料的型式是非常重要的，因爲這樣子，您才可以在您的程式中適當的處理數字的及文字的資料。

2.5.1 數　字

在討論變數(variables)之前，我們先研究一下 Apple 處理數字的格式。如果您目前處於整數 BASIC 情況下，數字只能以整數的型式出現，並且它們的範圍處於－32767與32767之間。

至於在 APPLESOFT][的情況時，數字則可以用整數、實數或科學表示(scientific notation)的方式表現出來。下面就是一些例子：

數　字	輸出格式
+1	1
-1	-1
3678	3678
-35.678	-35.678
45.72E5	4572000
1*10^20	1E+20
-12.34567896*10©10	-1.2345679E+11
1000000000	1E+09
999999999	999999999

圖2-3　資料型式

2.5.2 字　串

字串是資料型式的一種，一個字串是由許多的 ASCII 字所組成的；它們與數字最大的不同點就在於它們不能做算術的運算；一般說來，我們將一個句子放在引號中間就成爲一個字串了。下面就是一些字串的例子：

```
"HAVE A NICE DAY"
"I AM LEARNING TO PROGRAM"
"THIS IS A STRING"
""
```

字串可以由任何字組合而成，但是它的長度最多不能超過255 。

在Applesoft BASIC中具有許多處理字串的指令，我們可以使用它們進行各種字串的運作。

2.5.3 變 數

在設計 Apple 上使用的程式時，您可以使用三種型式的變數；一個變數就是一個名字，這個名字可以代表不同的數字或是字串；例如： X=1 ，X%=2, X$="GOOD" 等等。變數的型式也分爲實變數、整數變數以及字串變數。注意：在整數 BASIC 中您只能使用整數變數及字串變數。

在APPLE][中，變數名稱的長度可以有238個字，但是開頭必定是一個英文字母；雖然變數名稱可以有238個字，但是APPLE][只使用前面的兩個字代表這個變數。因此，APPLE][將把BOSQUE與BOSTON視爲同一變數(BO)，另外一個需要注意的是，在變數名稱內不可以包含有保留字(reserved words)，您可以參看附錄F，裏面有Apple II保留字的表列。下面是一些合法及不合法的變數名稱：

合法	不合法
NX	NEXT（不能使用保留字）
JC%	BE BAD （不合法的空白）
LZ$	FORTUNE（FOR是一個保留字）

圖2-4 合法及不合法的變數名稱

在 Applesoft中，整數變數的名稱後面跟著一個百分符號(%)，而這個變數所代表的值只能是整數而且不能超過允許的範圍。

至於字串變數的後面則接著一個錢號($)，它能夠代表0到255個字的字串。

實變數則就是用它的名字代表，它所能代表的實數範圍是正負值的9.99999999 E + 37。

變數的組成方式有三種：簡單式，向量式及矩陣式。

下面就是一些簡單式變數的例子：

```
X%=1
X =1.23
X$="1.23 WIDGETS"
```

一個簡單式的變數一次只能擁有一個數值。

第二種型式就是向量式或稱爲表列式。當您需要將資料存在一個變數內，如 NR，而您又想使用這個名字儲存五個不同的數字。請參考圖2－2，我們在這裏重複它的部份程式。

```
20 DIM NR(5)
30 FOR I = 1 TO 5:READ NR(I):NEXT I
32 DATA 5,9,11,4,3
40 MX =-1
```

NR 這個向量在行號 20 的地方被宣告，而實際上它擁有六個儲存元素，這些元素分別編號爲 0 到 5 。在這個例子中，我們使用的是元素 1 到元素 5 ；資料就如下面一般地被儲存起來；

```
 0  1  2  3  4  5  = 元素
--------------------
 0  5  9  11 4  3  = 值
```

當 FOR...NEXT 迴圈被執行時，脚註（ index ）I 就每次被增加 1（由 1 開始到 5 結束），因此每一個元素都被存入了 NR 這個向量內了。注意，元素零內的值是零，這是因爲在 BASIC 語言中，當它遇到了 DIM 敍述時，它會將向量或矩陣內所有元素的值變爲零，同時在 BASIC 中也允許您使用元素零，但是在 FORTRAN 或 COBOL 中並不具有這種功能。

在使用向量時，我們可以將向量視爲一個一階的矩陣。

第三種型式的變數就是矩陣變數；您這時候必定會問什麼是矩陣變數？一個矩陣就是將元素按照行及列安排的表。這些元素可以是實數，整數或是字串；下面就是一個實數矩陣的例子：

10.50	20.6	35.75	34.90
33.13	12.43	98.33	99.00
21.00	98.22	98.33	12.45
11.87	1.23	8.23	88.88
22.19	13.09	1.32	3.45

圖2-5 數字的矩陣

如您所見到的，上面這個矩陣擁有五列及四行的數字，數字1.23 就位於第 4 列，第二行的位置。如果這個矩陣要被放入 NR 這個變數內，那麼 1.23 這個數字就必須使用 NR(4,2) 的方式表示出來。 圖 2 － 6 就是將上面這個矩陣的值存入 NR 內的程式。

```
10 DIM NR(5,4)
20 FOR I= 1 TO 5
30 FOR J= 1 TO 4
40 READ NR(I,J)
50 NEXT J:NEXT I
60 DATA 10.5, 20.6,35.75,34,9
70 DATA 33.13,12.43,98.33,99.00
80 DATA 21,98.22,98.33,12.45
90 DATA 11.87,1.23,8.23,88.88
100 DATA 22.19,13.09,1.32,3.45
110 END
```

圖2-6 範例程式

2.6 變數值的設定

變數值的設定可以使用許多種不同的方法：您可以使用輸入敍述（ input statements），或是設定敍述（ assignment statements ）設定變數的值。不論您使用的是那一種方法，設定的工作都必須在程式中

進行；同時，您必須記得的是，變數被設定值時，值的資料型式必須與變數的型式相同。也就是說， NX% 這個整數變數無法完整地接受1.234這個值， NX% 只會接受1，而不會給您任何的錯誤訊息。但是如果實變數 HY 想要存入 "TOM" 這個值，那麼您就會得到錯誤訊息了。

一個設定敍述就是將變數的值設定。圖 2 － 7 就是一些合法的設定敍述。

```
LET  X = 45.64
LET  A$= "NOW IS THE TIME"
     Y = 2*5
LET NR = 2
```

圖2-7 設定敍述的例子

在設定敍述中LET是可以被省略的；有一個重要的概念必須牢記在心；在設定敍述中的等號（ = ）並不表示相等的意思，而是說將等號右邊的值放入等號左邊的變數內；這對初學者往往是較難瞭解的，但是各位必須設法適應它，因爲在一個程式中，設定敍述往往佔了程式80 %的部份。

由於有以上的規定，因此我們就可以設定 CR 變數的值爲 CR = CR + 1；這個式子表示，每當這個敍述被執行一次之後， CR 內的值就被增加了。

2.7流程的控制

在 2.2 節中我們曾經提過由上往下（ top-down ）設計程式的方法，但卻沒有確實地描述它；現在我們將詳細地討論這個方法。

在這種情形下，您由問題開始著手，然後將這個問題再轉變成確實的敍述；接著您才由上往下的設計一段一段的程式。

實際的過程包含了資料型式的宣告（在 2.5 節討論），程式的設計及結構（在 2.4 ， 2.6 及 3.2 節討論）等等工作。

2.7.1 由下往上(Bottom-up)的方式

這種型式的程式設計由一個簡單的程式或函數開始，而逐漸建立起複雜的函數或程式段，直到解決問題的程式被建立完畢時才停止。

這種方式的程式設計的過程是您必須先建立小的程式段(sections)，再將這些段結合成較大的程式片段(segment)，最後才將這些程式片段組合成完整的程式。由下往上的方式必須特別小心地去發展程式及注意程式的完整性。

2.7.2 由上往下(Top-Down)的方式

這種方式是將問題分解成許多個較小的部份，然後將這些部份組合起來找出最簡單的方法，使得設計出來的程式易於瞭解，同時設計的工作也較為簡易。

這種解決問題的方式是將問題分為許多片段，再將這些片段分為更詳細的段；最後才將這些段寫成程式行。

假設有一個問題P，它是由許多的片段及段所組成的；那麼我們依照由上往下的結構方式可以繪出下面的圖。

```
                        P
                        !
           +------------+------------+ ...
           !            !            !
           P1           P2           P3
           !            !            !
       +---+---+      +-+-+      +---+---+
       !   !   !      !   !      !   !   !    ..
      P11 P12 P13    P21 P22    P31 P32 P33
               !
             +-+-+
             !   !     ...
           P131  P132
```

圖2-8 由上往下的邏輯結構

上面這個圖並非是流程圖（flowchart ），而是一個邏輯樹（logic tree ），它表示出如何將一個問題分成許多個段或片段，而由這些片段或段又將整個問題組合在一起；由這個邏輯樹您就可以描述出這個問題的邏輯及流程圖了。我們可以由這個圖中立刻知道，問題 P 是由 P1-P3 的副問題（sub-problems ）所組成的，而每一個片段又由許多個段所組成；例如，副問題 P1 就是由 P11-P13 的段所組成。

2.8 輸入

資料是由輸入的敍述輸入到電腦裏面的；只要是能夠將資料輸入到電腦內的敍述，我們就稱它爲輸入敍述（input statements ）。它們分別是 input敍述，GET 敍述及 DATA 敍述，我們將在下面討論它們。

2.8.1 INPUT敍述

在 BASIC 中的一個輸入敍述就是 INPUT 敍述；這個敍述允許您由鍵盤輸入資料，而不需要將資料全部先在程式中設定。圖2 － 9就是一個使用了 INPUT 敍述的程式。

```
10 X=5
20 INPUT Y
30 Z=X * Y
40 PRINT Z
50 END
```

圖2-9 input的例子

當這個程式執行到行號 20 的地方時，程式的執行就會暫時停止，並且您將在螢光幕上看到一個問號（？），等待著您輸入Y的值；只要您輸入了一個值之後，這個值就會被放入Y內，而這個程式就繼續往下執行了。

INPUT 敍述可以像 print 敍述一般地將文字資料包含在引號之間使

用。行號20可以改為：

```
20 INPUT "ENTER THE VALUE YOU WISH FOR Y: ";Y
```

電腦這時將會把“ ENTER THE VALUE YOU WISH FOR Y: ”這個訊息顯示在螢光幕上並等待著您輸入Y值。

2.8.2 GET敍述

GET 敍述與 INPUT 敍述相同的地方就是它們同樣地等待您由鍵盤輸入資料，不同點則在於 GET 敍述只接受一個字的輸入。一般的情況下，我們只有在命令電腦等待著我們由鍵盤輸入一個字後才繼續程式的執行時才使用 GET 敍述。圖 2－10 就是一個使用 GET 敍述的小程式。

```
    :
20 INPUT "YOUR NAME HERE:";A$
30 PRINT " PRESS ANY KEY TO CONTINUE "
40 GET B$
50 PRINT:PRINT
60 PRINT A$
70 END
```

圖2-10　Get的例子

在這個程式執行時，您在輸入了名字之後，您將見到 "PRESS ANY KEY TO CONTINUE" 的訊息，而電腦這時就在等待著您按下一個鍵，然後程式才能繼續執行。

2.8.3 DATA敍述

如果一個程式在每次執行時它所需要使用的資料都相同，那麼這些資料就可以使用 DATA 敍述將它們存在程式中，而您在使用時也就不必每次重新輸入了。 DATA 敍述可以放在程式中任何的位置，也只有在 READ 敍述被執行時，DATA 內的資料才會被使用。否則，DATA 敍述將不被理會。一個 DATA 敍述內可以包含不只一個的資料，資料與資料之間必須使用逗號分開

，並且只要不超過255個字（包含空白、逗號、行號、DATA 等等）的資料都可以使用一個 DATA 敍述輸入。您可以參看圖 2 － 2 的例子。

在 DATA 敍述中的資料是按照順序由 READ 敍述取用的。當電腦執行程式遇到 READ 敍述時，電腦就會往 DATA 敍述內由第一個資料開始取用，接著是第二個，接著是第三個等等。如果在程式中您希望再由 DATA 敍述中的第一個資料開始讀起，那麼您就必須在程式中使用 RESTORE 命令了。

2.8.4 顯示訊息

大多數的程式都是需要由使用者輸入資料而不是將資料包含在程式內的；因此，當要求使用者輸入資料時，就必須將訊息通知使用者，以便他（或她）能夠輸入正確的資料。

假設，現在有一個程式要求使用者輸入目前的日期；當然，有許多種方法完成這個工作，但是為了要使印出或計算方便起見，這個程式希望輸入的日期是依照某一種格式輸入的；若是輸入格式並不正確，那麼在程式中的處理工作可能就會發生錯誤了。

INPUT 敍述可以與行號共同使用，這使得使用者在輸入日期之前就能夠先知道輸入資料的格式，而不至於發生錯誤；假設我們希望這個程式要求的輸入日期是以月，日，年的次序排列的，並且在它們中間都以一條斜綫分開；那麼我們就可以在 INPUT 敍述中使用訊息通知使用者輸入日期的格式了，就如同：

```
20 INPUT "ENTER DATE: MM/DD/YY ";DT$
```

那麼對使用者而言，他（或她）就知道輸入像 05/05/81 的日期了，而這個日期就會被放入字串變數 DT$ 內。

2.8.5 螢幕顯示

當我們使用訊息通知使用者時，我們必須知道螢幕顯示的情形；第一個

考慮的情形必定是要將螢幕清除乾淨，我們使用的就是 HOME 命令（當電腦處於圖形狀況下時，就必須用 text 命令），這個命令就會使螢幕被清除乾淨並且將游標（ cursor ）移到螢幕的左上角位置。圖 2 － 11就是一個要求輸入日期及使用者名字的程式。

```
10 DIM N$(2)
20 HOME
30 INPUT " ENTER TODAY'S DATE: MM/DD/YY ";D$
40 INPUT " ENTER YOUR NAME: LAST, FIRST
   ";N$(1),N$(2)
   :
   :

100 END
```

圖2-11 顯示訊息的例子

在您對這個程式的顯示訊息做了正確地反應後，您將看到

```
ENTER TODAY'S DATE05/05/81
ENTER YOUR NAME LAST, FIRSTBLOW,JOE
```

這並不是一個令人覺得悅目的顯示格式；在圖 2 － 12中我們重新設計了一個新的顯示格式使您覺得愉悅一些。

```
10 DIM N$(2)
15 HOME
20 PRINT:PRINT:PRINT
30 HTAB 15:INPUT "ENTER TODAY'S DATE: MM/DD/YY ";D$
40 PRINT:PRINT:PRINT
50 HTAB 15:INPUT "ENTER YOUR NAME: LAST, FIRST
   ";N$(1),N$(2)
   :
   :

100 END
```

圖2-12 較好的顯示訊息

2.9 輸出

在考慮輸入資料的同時，我們也必須對資料的輸出做一番研究，不論您想將資料輸出到螢幕上或是到印表機（ printer ）上，我們都要對輸出的敍述加以認識。

2.9.1 PRINT敍述

BASIC 與電腦使用者最主要的聯繫方法就是使用 PRINT 命令，您可以參看附錄A及B，裏面包含了 PRINT 的各種格式。使用 PRINT 敍述顯示資料或是印出資料有許多種不同方法，如同您在圖2－2的例子中見到的 print敍述，它只是跳過一行顯示行的位置而不顯示出任何訊息；請記住一點，那就是包含在 PRINT 敍述中引號內的訊息都會原封不動的顯示出來；因此，如果您想將一個變數的值顯示出來，那麼您絕對不可以將這個變數包含在引號之內。下面是一個例子：

```
   :
10 A = 10
20 PRINT "A"
30 PRINT A
   :
   ]RUN

   A
   10
```

2.9.2 設定格式

訊息在被輸出之前最好先被設定格式，這使得使用者較易瞭解輸出資料的意義。如果現在有一個程式要將變數 W（表示 widgets ），及變數 T（表示 thingamajigs ）顯示出來，這個程式可以設計爲：

```
40 PRINT W: PRINT T
```

假設螢幕已經被清除乾淨了，同時W及T的值分別是3與4，那麼您將在螢幕上看到：

```
3
4
```

這樣的顯示情形使得使用者難以瞭解電腦表示的是什麼意思；因此我們可以使用下面圖2－13的程式將顯示資料的意義說明清楚。

```
40 PRINT
50 PRINT "     WIDGETS ON HAND:    ";W
60 PRINT
70 PRINT "THINGAMAJIGS ON HAND:    ";T

]RUN

     WIDGETS ON HAND:    3

THINGAMAJIGS ON HAND:    4
```

圖2-13 Print的例子

2.9.3　螢幕顯示的安排

我們已經討論過了如何使訊息顯示在螢幕上而使人覺得易於瞭解的技術；同樣的，在做輸出工作時，我們也要考慮到這個問題。

每一個值，不論是數字的或是文字的，都應該在螢幕上被顯示在令人易於瞭解的位置。

數字在顯示時應該向左看齊；如果這個數字代表錢，那麼在這個數字之前應該加上錢號；至於其他的細節您可以參看第七章的PRINT USING程式。

當螢幕被填滿時應該往上捲讓出空位使得隨後的資料得以有空間顯示出來，而往上捲的方式最好是由頭到尾一次捲完，使得整個畫面看起來整潔而

完整。幸運的是，APPLE 本身就能夠做這個工作，只要您使用 VTAB，，HTAB， CALL -958，及 CALL -868（參看附錄C）指令就可以了；同樣地，您也可以使用 FOR...NEXT 廻圈控制這個動作（參看第三章）。

2.10 處理過程

處理過程包括了解決問題的動作，這個動作是由您將所有問題的分析組合起來，並將程式依照正確地程序（ algorithms ）設計完成，再使用這個程式將問題解決。

處理的過程同時也是代表在輸出程式段之前的程式主體，它一般就是程式中最主要的部份或是一個主要的副程式（ subroutine ）。

2.10.1 程 序

程序就是一連串的敍述依照順序執行，而能夠產生您所需要的結果；程序必須有五種特性，那就是：定義完善，並非一直執行永不停止的，具有輸入與輸出的功能，及具有解決問題的有效性。我們也可以在程序中加入說明，使得我們能對程序及程序的執行更易於瞭解。有關程序的例子有：數學的步驟，烹飪術，以及如何修理漏水管的指示等等。

程序必須被翻譯成適當的語言，使得使用此程序的人能夠輕易的瞭解；需要記住的一點就是，一個程序是爲了使讀者能夠加深他對問題及解決方法的瞭解；而非對電腦。因此，所有的程序都必須被轉換成適當的程式語言，以便能夠利用電腦解決問題。

一個程式就是使用某種程式語言的一個程序；因此，重要的一個關鍵就在這裏，由於我們先有程序，而程序是以文字說明的方式設計的，它往往由許多的敍述，算術運算等結合而成；因此，我們在利用電腦解決問題時，就必須將程序使用適當的程式語言表示出來。

假設您想將向量中兩個元素的值互換，那麼下面就是使用 BASIC 語言設計的程式：

```
40 DIM V(20)
50 T = 0
     :
     :

1150 T = V(14)
1160 V(14) = V(15)
1170 V(15) = T
     :
     :
```

圖 2－14 程序的例子

2.10.2 修正錯誤

想要確定一個程式，尤其是一個很大的程式，是完全正確的，往往是一件費時且困難的工作。

在我們設計程式時，往往會有許多的錯誤發生，因此只要我們想使用電腦解決問題，那麼我們就必須接受這個事實。一般來說，設計程式一半以上的努力都將花在尋找出程式的錯誤及修正這些錯誤的工作上。即使我們認爲已經修正完畢了之後，程式中仍然存有許多未被發現的錯誤，而這些錯誤可能要在使用了許多次之後才被發現。由於有這種情形的存在，使得目前使用電腦的人們對於電腦都缺乏足夠的信心。但是有一點我們必須確認的，那就是電腦發生錯誤的機會非常小，發生錯誤的原因大都是由於程式設計的不正確或是未能適當地完成測試。

下面有幾階層除錯的過程，對於設計程式的人而言必須要考慮到的。

1. 在程式中已經沒有語法（ syntax ）的錯誤了。
2. 在程式中已經沒有結構上的錯誤了。
3. 這個程式對測試的資料能夠正確地工作。
4. 這個程式對所有有效的資料都能正確地工作。
5. 這個程式對所有可能的資料都能正確地工作，並能符合問題的要求。

6. 這個程式對所有的資料都能正確地工作，能不接受無效的資料，並能產生正確的答案。

幾乎所有的程式都能達到第 4 階層的正確性，目前我們所見到的程式也能達到第 5 階層的要求；至於第 6 階層則往往要等到程式在市場上銷售了一陣子之後才能確定。然而，對於微電腦而言，這個階層是很難達到的；這是因爲以下兩個原因：

1. 微電腦硬體方面的限制；記憶體的大小及程式語言無法有效的應用。
2. 程式設計師不願在設計程式時花費足夠的心力及時間在他（或她）的工作上。

由於目前微電腦快速發展的事實，第 1 個原因似乎已經不成爲原因了；因此只有第二個原因才是眞正問題的所在。

只要程式設計師願意花下時間及努力，那麼相信第 6 階層的程式將會在市場上出現的。

2.11 設計程式的一般規則

在這一節中我們將告訴您一些規則，如果您在設計程式時能夠遵循它們的規定，那麼必將使您的程式錯誤減少並且能夠產生出正確的結果。

1. 您在任何時間都是冷靜的。
2. 您對問題的定義非常詳細、完整。
3. 您使用的是由上往下的方式。
4. 您能夠設計程式的結構。
5. 您能夠使用正確的語法。
6. 您能夠在開始的時候就做記錄的工作。
7. 您能夠專心的致力於問題上。

8. 您設計程式時能合乎邏輯。
9. 您的程式中儘量能夠少用 GOTO 命令。
10. 您在程式中能夠多用 GOSUB。
11. 您使輸出的資料令人能夠容易的接受。
12. 您不會故意的不去遵從一般的規則。

2.12 總 結

在這一章中，我們對於電腦的一般概念及 APPLE 的詳細情形都已經討論過了；電腦是以方塊圖的方式解說給您看的；至於每一個元件我們只是略爲介紹罷了，那是因爲本書的目的並不在於硬體的結構。

我們對於計算的過程做了較詳細地討論；我們是由基本問題一直研究到問題的解決，相信這是一種較只是坐在電腦前面任意去創造爲佳的方法。

至於選擇問題、描述問題、設計解答，用程式語言設計出來，確定程式的正確性，以及對程式做好紀錄的工作，我們都是使用範例解說給您的。

至於較深一層的介紹則就是 APPLE 的 BASIC 語言，資料的型式，及變數的名稱與需要注意的地方。

流程的控制－－不論是由上往下或是由下往上－－我們都做了介紹；同時發現，對我們來說由上往下似乎是一種較好的方式。

在接近本章結束的部份，我們介紹了設計程式時需要注意的幾個問題，相信您現在已經可以設計您自己的小程式了。我們希望您能夠多使用機器做您自己的練習，這對您的幫助是很大的。

最後一節，2.11 ，討論的是一般的規定，您在往後設計程式時，必須確實遵照規定，否則您將無法得到預期的效果。

第3章

程式的發展

3.0 大 綱

在這一章中，我們將討論程式的一般組織情形，使您能夠對電腦程式的主幹及細節加以瞭解，並認識它們的組合方式。

相信您曾經多次的聽人說過“管它如何，只要這個程式能夠正確執行就可以了！”這樣的話；當然，一個程式能夠正確地依照我們的需要工作是我們所追求的目的；然而，當我們需要改變、增加或減少這個程式的內容時，又當如何呢？

如果輸入錯誤的資料，或由新手操作這個程式時，是否這個程式仍然能夠正確地執行呢？

這一章就是專注於討論這些問題，同時考慮的範圍還有速度、整合性、管家等等的工作。

我們希望您在讀這本書時能夠輸入書中的程式，執行並研究它們，而使您在讀完這本書後能夠設計出更好的程式，得到更好的結果。

3.1 組織方式的考慮

我們由組織的觀點來研究程式的發展；在這一節中，我們只是大略地描述這個概念，因爲我們將在其他的地方繼續地討論這個問題；在這裏只是先讓您熟悉一下罷了。

3.1.1 速　度

在一般的情況下，微電腦執行程式的速度對大多數的應用而言都是足夠快的了；但是，當您需要等待電腦給予您一個回應以便做決定時，那麼速度似乎就成了一個值得考慮的問題了。大多數的情形，您都可以改變您的程式而使程式執行的速度加快；程式執行速度較慢的原因往往是因爲程式的結構或是表示變數方式的不正確而導致的。

第一件我們要注意的就是數字的表示方法；如果一個常數使用在 FOR-NEXT 廻圈內，並且被重複許多次，那麼您最好將它移出廻圈並且用一個變數代替它，這將使得廻圈執行的速度加快許多。這是因爲常數在被使用之前必須先被轉換成浮點（ floating point ）的型式，而浮點值的轉換工作非常浪費時間；因此，對於廻圈中的常數最好先在廻圈之外設定，以免每次執行廻圈一次就要做一次轉換的工作。圖 3 － 1 就是改變的情形。

第二個重要的項目就是變數的放置位置問題。變數是依出現的順序存在變數表（ variable table ）中的。例如，下面這個敘述：

```
10 X = 0: Y = 5: Z = 15
   :
1000 END
```

```
100 FOR I = 1 TO 1000
110 PI = 3.1415926
120 C = 2 * PI * 12
130 PRINT C
140 NEXT I
150 END
```

(a)

```
80  R = 12
90  C = 2 * 3.1415926 * R
100 FOR I = 1 TO 1000
130 PRINT C
140 NEXT I
150 END
```

(b)

圖3-1　移動變數位置的例子

變數 X 第一個被存入表中，接著才是 Y ，最後是 Z 。當您將一個值存入一個

變數中時（如 X=Z ），電腦就會在表中尋找X及Y的位置，等到找到Z的位置之後，X的值就會被移往Z的記憶體位置內。因此，如果Z時常被使用，那麼最好是將它存在變數表前面的位置，這樣子就可以縮短尋找的時間。當您使用元素數目較多的二階或三階矩陣時，這種考慮顯得更爲重要。

還有一種增加速度的方法就是在 NEXT 敍述後面不使用變數元素；這也就是說像上面例子中的140行號的敍述改寫爲 140 NEXT，但是這麼做可能產生兩個問題；第一個就是 APPLESOFT 無法知道所執行的 NEXT 是否爲最近被執行的迴圈；第二則是當您在偵錯時可能增加困難。

最後一個對速度有關的因素我們將在本書的其他部份中再度的討論；這個因素就是直接的轉移命令（如 GOTO 1505 ）或是副程式的呼叫（如 GOSUB 55 ）。當我們在程式中要求處理某一個行號的程式行時，APPLESOFT 就會從這個程式的起始行號開始往下尋找這個指定的行號，直到被發現時爲止。因此，如果我們呼叫的副程式是經常被使用到的，那麼如果它的起始行號較小，尋找這個副程式的時間就得以節省許多了。

3.1.2 整合性

這一節討論的是變異性，它如何對一般的程式設計及 Apple 電腦發生影響；討論是針對於速度及程式的簡潔而考慮的。同時我們也將主程式與副程式的重要性告訴您。

3.1.3 主程式

主程式就是一個程式中最主要的控制邏輯，也就是在這個範圍內呼叫副程式或將執行轉移到其他的部份；但是，我們需要瞭解的一點就是程式的邏輯應該一定要回到主程式。

主程式控制所有程式中的動作，如輸入資料，螢幕格式的控制，輸出訊息，程式的進行，外圍設備的控制，以及程式的結束與開始。

主程式的工作就是要將問題依照合乎邏輯的方式解決；因此對於主程式的位置必須加以考慮，我們在第四章中將討論主程式的位置問題（以行號爲準）。

至於程式中其他部份間互相的關係也就提供了程式中必須的某些功能。在 2.2.1 節中的圖就是一個只有主程式的程式，因爲在這個程式中並不需要任何其他的功能。

3.1.4　副程式

這一節討論副程式，副程式可以說是與主程式互相依賴，同時又提供程式中許多必要功能的元素。同時我們要討論的是它在程式中的位置，主要的考慮因素則是它在程式中提供的功能。

有時候在一個程式中的某一個部份必須在不同的地方重複地被執行許多次，那麼如果我們在每一個必須被執行的地方都重複將這個程式段編寫進去，那豈不是非常費時且費事嗎？爲了要避免這種情況，因此我們就使用一個可以代表這許多相同程式段的副程式，並把它放在程式中的某一個位置，當我們要使用它時，只需要由主程式中“呼叫”它就可以了，在副程式被執行完畢之後，控制又會回到主程式繼續執行了。

副程式的功用就在於節省程式的空間，並且節省重複寫程式行的工作，但是它並不能節省程式的執行時間，至於進入及離開副程式的時間對於整個程式的執行時間而言似乎並不是很重要的。

副程式還有許多好處；第一就是它使您對於整個程式的結構能夠較容易的瞭解；當您的程式發生了問題的時候，副程式就使您的偵錯工作不致於太困難，這是因爲整個程式的邏輯都已經被分散在副程式中的原因，問題很快地就可以被獨立出來而改正了。

由於變數表及字串表都是被存放在一個固定的區域內，因此在 BASIC 語言中使用副程式是非常適當的；所謂固定區域的意思就是程式可以直接地去處理存在這個區域內的值，而不必再經過轉換的工作；至於其他的語言，則因爲表示位置的方法有許多種，因此在主程式與副程式之間做傳送資料的工作時，就必須詳細地說明； Apple ，因爲它是一個個人電腦，並且具有固定區域型式的記憶體；因此，我們使用副程式是非常簡易的。

當您的能力增加了之後，您將可以發展一系列的副程式－－我們可稱它們爲一個 library ，並把它們儲存起來，當其他的程式需要使用它們的時候，您就可以將這些副程式插入其中，共同使用了。這個插入的工作，您可

以使用 DOS 3.3 System Master磁片中的 RENUMBER 程式達成目的。

副程式結構中最簡單的就是一個進入點，一個離開點的程式，您所設計的副程式大部份都將是這種型式的。然而，當您設計的副程式必須依照許多的元素值而做決定，並且每當這個副程式被呼叫時，都必須依照這些值而做相對應的反應的時候；這就使得程式設計的工作需要配合各種情況的設定而做反應了。

圖 3 － 2 就是一個一般副程式的結構。

圖 3 － 3 則是一個有許多個進入點的副程式。

當一個副程式被呼叫（ GOSUB ）之後，在程式中的下一個敍述的位置就被編譯程式（ interpreter ）儲存起來，這使得副程式在被執行完畢之後，能夠正確地回到主程式中適當的程式行繼續程式的執行。

在圖 3 － 3 的例子中顯示了傳送元素到副程式中的情形，但在事實上，這只是一個模擬而非眞實的情況，因爲它是在進入副程式之前就先定義了變數值。由於我們使用的電腦的記憶體是固定區域的型式，因此傳送副程式的元素就是在呼叫它之前先定義這個元素值的意思。

注意：以上的討論只適用於 Apple 機器上的 BASIC 語言，至於其他的語言對於副程式的安排又有不同的做法。

3.2　程式的結構

在 BASIC 語言中——事實上是所有的程式語言，都具有許多種的程式結構，使得我們在設計程式時能夠儘量的發揮電腦的能力，並且使您能夠設計出結構完善的程式來。

我們現在將要討論的四種基本結構就是順序處理（ SEQUENCE ），FOR...NEXT 或 DO-UNTIL ，CASE，及 IF-THEN ELSE 。

基本的結構我們可以藉著流程圖來說明，這將使我們對設計的過程能夠以較容易瞭解；簡單地說，程式中必須使 GOTO 命令出現的次數減到最少，同時程式能夠由上到下輕易的使人瞭解，並且看起來是令人愉悅的。除此之外，尙有一點非常重要，那就是依照這種方式設計的程式能夠只有一個進入點，而且是只有一個結束點。

```
    :
    :

50 GOTO 900
51 REM **************
52 REM * GEN'L SUBR *
53 REM **************
    :
    :

99 REM * COUNT UP
100 FOR I = 1 TO 10
     :
     :

150 NEXT I
160 RETURN
     :
     :
199 REM * COUNT DOWN
200 FOR I = 10 TO 1 STEP -1
     :
     :

250 NEXT I
260 RETURN
     :
     :
899 REM MAIN DRIVER
900 GOSUB 100
910 :
     :
     :

1000 GOSUB 100 <---     呼叫副程式並把返回
1010 :                  的位置儲存起來
      :
      :

2000 GOSUB 200
2010 :
      :
      :

5000 END
```

圖3-2　副程式的結構

```
      :
      :

50 GOTO 1000
51 REM **************
52 REM * GEN'L SUBR *
53 REM **************
      :
      :

99 REM * MULTI-ENTRY SUBR
100 M = 1
110 S = 1
120 M = M * 2
130 R = M * S
140 P = R + 2
150 RETURN
      :
      :

999 REM MAIN DRIVER
1000 :
     :

1050 GOSUB 100
1060 PRINT P
       :
       :

1100 M = 4:GOSUB 110
1110 PRINT P
       :
       :

1200 M = 2:S = 3:GOSUB 120
1210 PRINT P
       :
       :

1300 R = 5:GOSUB 140
1310 PRINT P
       :
       :

2000 END
```

圖3-3 多重進入點的副程式

3.2.1 順　序

假設現在我們要解一個問題——P，而P是由兩個部份F及S所組成的，F代表第一部份，S代表第二部份；因此，P = F + S。在圖3－4中的流程圖說明了這個狀況，由於F及S都是程式段，因此它們就成為了處理塊（ processing blocks ），並且依照順序分別處理。圖3－5就是依照3－4的流程圖而設計的程式。

```
        ***********
        *         *
    -->*   問題    *-->
        *         *
        ***********
```

(a)

```
    *********     **********
    *       *     *        *
-->*第一部份*-->*  第二部份 *-->
    *       *     *        *
    *       *     *        *
    *********     **********
```

(b)

圖3-4　順序處理

```
100 REM * FIRST PART
        :
        :
200 REM * SECOND PART
        :
        :
300 END
```

圖3-5　範例程式

對於順序而言並沒有什麼神奇的，它只是一連串處理的步驟，並且依照順序執行。然而，一個順序也可以包含其他的結構，它可以包含一個完整的邏輯程式段，而這個程式段所執行的功能就是產生一些特定的結果或是做某一項特定的工作；當然，它也可以有許多層屬於自己的結構。

3.2.2　FOR-NEXT迴圈

FOR...NEXT 的組織情形可能是在 APPLESOFT 及整數 BASIC 中最常被使用的組織了；這樣的組織允許您重複地執行同樣的迴圈，這樣的組織事實上就是 DO-UNTIL 的型式，您重複地執行同樣的動作直到某個情況產生了為止。它的整個邏輯就如圖3－6一般；假設您有一個函數F，想要被執行若干次，而您設計的程式中包含了一個變數計算執行這個函數的次數，當這個計數用的變數達到了某一個極限值之後，這個過程就結束了，而程式就繼續往下執行。

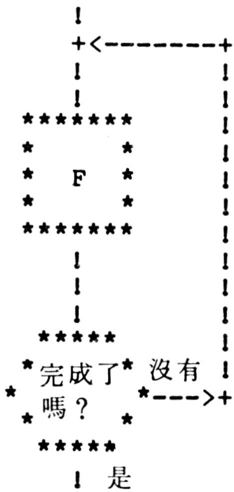


圖3-6　Do-until的邏輯

注意一點，那就是 DO-UNTIL 的結構至少將F函數執行一次，而 FOR

```
                  !
                  !
             *********
             *       *
             * 設定脚 *
             * 註值   *
             *       *
             *********
                 !
             *********
             *       *
             *       *
             * 設定範圍 *              I
             *       *
             *       *
             *********
                 !
             *********
             *       *
             * 設定増 *
             * 加值   *
             *       *
             *********
                 !
  +---------->+
  !              !
  !         ***********
  !         *         *
  !         *         *
  !         * 處理部份  *             II
  !         *         *
  !         *         *
  !         ***********
  !              !
  !         ***********
  !         *         *
  !         * I=I + S *             III
  !         *         *
  !         ***********
  !              !
  !            *****
  !          *       *
  ! 不是    *   IS     *
  +<---*    I > R    *            IV
        *    ?     *
          *       *
            *****
              !  是
```

圖3-7 For…Next迴圈的邏輯

FOR...NEXT 這個命令可能是 BASIC 中最具功效的功能（除了副程式之外），它是由四個部份所組成的，它們分別是；

I 起始段（設定初值）
II 執行段（執行程式）
III 增加段（增加變數值）
IV 測試段（做測試工作）

這些段都在圖 3 － 7 中做了說明，第一段事實上有三個變數，這些變數必須先被設定初值；這個指令的一般格式是：

FOR 腳註變數 = 起始值 TO 終止值（STEP 增加值）

例如：FOR I = 0 TO 15 STEP 3

第一件要注意的事就是括在括弧內的項目，STEP 及增加值是可以被省略的；如果您在這個敍述中不指出 STEP 值，那麼 1 就被設定爲增加值；在這種情況下，FOR...NEXT 這個廻圈被執行的次數就每次增加 1（由起始值開始）。至於其他的參數都是必須在這個指令中被指明的。

腳註變數（ index variable ）就是用來計算 FOR...NEXT 廻圈被執行的次數的；起始值及終止值也就是腳註變數的起始值及結束值。

至於增加值的部份可以有負值與正值兩種情況，這都是由 STEP 值而決定；每一次廻圈執行到 NEXT 命令時，就會依照 STEP 值而增加或減少腳註值，這個動作是在測試 FOR...NEXT 廻圈是否執行完成了的同時從事的。

最後的部份就是測試了，它發生在腳註值被增加之後；測試的工作就是測試腳註值是否較結束值爲大，如果腳註值並不比結束值大，那麼廻圈就將至少再被執行一次；如果腳註值較大，那麼廻圈就結束，而程式就繼續往下執行了。

請參看圖 3－8，3－9，2－2 這些合法的FOR...NEXT的組織情形。

圖 3 － 9 是一個巢狀的 FOR...NEXT 結構程式，所謂巢狀（ nested ）的意思是說一個 FOR...NEXT 的廻圈包含在另一個 FOR...NEXT 廻圈內。

```
     :

1000 FOR I = 1 to 10 ->+
     :                 !
     :                 !
                       !
1090 NEXT I <----------+
     :
1100 FOR J = 2 to 23 STEP 2 ->+
     :                        !
     :                        !
                              !
1180 NEXT J <-----------------+
     :
     :

1255 FOR X = 2.2 TO 4.4 STEP 0.1 ->+
     :                             !
     :                             !
                                   !
1444 NEXT X <----------------------+
     :
     :

1822 END
```

圖3-8 For…Next程式的例子

```
     :
     :
2000 FOR IX = 0 TO 4 --------------->+
2010 VTAB 4:CALL -958                !
2020 FOR J = 1 TO 5  ---------->+    !
2030 I = INT(IX * 5 + J)        !    !
2040 PRINT "ITERATION # = ";I   !    !
2050 NEXT J              <------+    !
2060 VTAB 20:INPUT "TYPE RETURN KEY  !
     TO CONTINUE ";A$                !
2070 NEXT IX             <-----------+
     :
     :

5000 END
```

圖3-9 巢狀的For…Next迴圈

圖 3 — 10是一個不合法的結構，我們用這個例子告訴您不可以有相交的迴圈，這是非常重要的，千萬不要在您的程式中發生這樣的情形。

```
     :
     :

1500 FOR K = 1 TO 25 ------------+
     :                           !
     :                           !
                                 !
1600 FOR J = 10 TO 1 STEP -1 --+ !
     :                         ! !
     :                         ! !
                               ! !
1700 FOR I = 1 TO 5 ---------+ ! !
     :                       ! ! !
     :                       ! ! !
                             ! ! !
1800 NEXT J <--------------- ! + !
     :                       !   !
     :                       !   !
                             !   !
1900 FOR L = 1 TO 4 -------+ !   !
     :                     ! !   !
     :                     ! !   !
                           ! !   !
2000 NEXT I <------------- ! +   !
     :                     !     !
     :                     !     !
                           !     !
2100 NEXT L <--------------+     !
     :                           !
     :                           !
                                 !
2200 NEXT K <--------------------+
     :
     :

2300 END
```

圖3-10 不合法的For…Next 結構

3.2.3 CASE的結構

CASE 的結構是一個多重方向的轉移（依據某些情況或某些值），它允許您依照某一個變數的值而做不同的轉移。

這個變數可以是一個算術運算式，在執行轉移之前先被計算。圖 3－11 就是 CASE 敍述的邏輯結構。假設您現在有幾種函數 F1,F2,...Fn，而這些函數都必須在某種情況下被執行；CASE 的結構就是允許您做這樣的工作。

```
         !  <-- 控制程式
       *****
      *     *
     *  exp  *    exp = 運算式
      *     *       e = 運算式的值
       *****
         !
         !      ******
         ! e=1  *    *
         +---->* F1 *---->+
         !      *    *    !
         !      ******    !
         !                !
         !      ******    !
         ! e=2  *    *    !
         +---->* F2 *---->+
         !      *    *    !
         !      ******    !
         !                !
         !      ******    !
         ! e=3  *    *    !
         +---->* F3 *---->+
         !      *    *    !
         !      ******    !
         !                !
         !      ******    !
         ! e<1  *    *    !
         +---->* F4 *---->+
           e>3  *    *    !
                ******    ! <--控制程式
```

圖3-11　Case的結構

這種 CASE 的結構您在 APPLESOFT 及整數 BASIC 中都可以使用，但是它

們的語法（ syntax ）有些不同，因此您必須對兩種型式加以瞭解。在 APPLESOFT 中 CASE 的結構是以下面的指令完成的：

```
ON 變數 GOTO L1,L2,L3,..... 或是
ON 變數 GOSUB L1,L2,L3,....
```

圖3-12 Case的程式段

L 表示您依照變數的值而做轉移的程式行的行號。如果變數值是 1 ，那麼程式就轉移到行號 L1 的部份去執行，如果變數值是 2 ，那麼程式就往 L2 的行號去執行。但是如果變數值是零或是較擁有行號數目為大的值，那麼程式就往下執行而不理會轉移的行號；這種情況在圖 3－11及第 4.7 節中都可以看得到。

3.2.4 IF-THEN-ELSE

IF-THEN-ELSE 是一個兩個方向的轉移結構，它使得一個程式能夠依照眞或假（ TRUE-FALSE ）的情況而做執行的轉移；當情況測試的結果是眞，那麼程式就往一個方向轉移（ THEN ），如果情況是假，程式則往另一方向（ ELSE ）轉移。

```
                      *********
                      *       *
               +---->*  THEN-  *----->+
               !      * 的部份 *      !
          眞   !      *       *       !
               !      *********       !
            *******                   !
           *       *                  !
     ---->*  IF-的測試*               +---->
           *       *                  !
            *******                   !
               !      *********       !
          假   !      *       *       !
               +---->*  ELSE-  *----->+
                      * 的部份 *
                      *       *
                      *********
```

圖3-13 If-then-else的邏輯

```
     :
     :

90 A = INT(RND(1) * 2 + 0.5)
100 IF A = 1 THEN HOME:VTAB 10:PRINT
    "TRUE CONDITION MET":GOTO 200
110 HOME:VTAB 10
120 PRINT "FALSE CONDITION MET"
    :

200 REM * CONTINUE CODE HERE
    :

300 END
```

圖3-14　If-Then-else-的範例

3.2.5　不停止的迴圈

不停止的迴圈（ forever loop ）事實上並不是一個可以被接受的程式結構，但是由於Apple 具有遊戲控制器（ game paddles ）及發聲器可以與外界聯繫，因此我們將這種結構包含在這裏討論。我們可以使用這種方式令電腦等待著直到我們由外界輸入了某一個訊號之後再繼續程式的執行。下面就是一個允許這種情形的程式段。

```
           *******
           *     *
       +<--*  F  *<-----------+
       !   *     *            !  眞
       !   *******            !
       !                  **********
       !                 *          *
---->+--------------->* WHILE-的測試 *--->
                         *          *
                          **********    假
```

圖3-15　Do-wheile的邏輯

```
0   REM ********************************
10  REM *
20  REM *  SET PDL(0) TO MAXIMUM VALUE
30  REM *
40  REM *  RUN PROGRAM
50  REM *
60  REM *  REDUCE PDL(0) SETTING SLOWLY
70  REM *  UNTIL LINE 130 IS TRUE
80  REM *
90  REM ********************************
100  FOR I=1 TO 4
110  I=2
120  X=PDL(0)
130  IF X < 128 THEN I = 4
140  NEXT I
200  REM * CONTINUE PROGRAM HERE
210  HOME: VTAB 10: PRINT "GOODBYE FO
      R NOW....": END
```

圖3-16 不停的迴圈

我們現在必須指出一點，那就是上面的這種結構在其他的BASICS中並不被允許，它只能在Apple 中使用；這是因爲變數表被存在固定區域內而且脚註變數並不會被編譯程式設定爲不同的碼；因此，您可以任意的處理脚註變數，但是請您對這種情況特別的加以留意。

上面這個程式的廻圈將一直停留在 FOR...NEXT 的結構之內，因爲脚註變數 I 的值不是 2 就是 3 ，只要遊戲控制器一直是大於 128 ，這個廻圈的脚註值就無法超過 4 ；當PDL(0) < 128時，脚註變數 I 的值才被設定爲 4 。

這時候 I 的值才等於終止值，而在測試的工作完成了之後，程式才能夠離開廻圈繼續往下執行；使用這種方式我們就可以模擬 DO-WHILE 的結構了。

圖 3 － 17是另外一個 DO-WHILE 的結構，在這個例子中我們將告訴您一個測試鍵盤的方法；在這個測試中您可以測試每一個鍵，並將您打入的字及與它相對應的十進位值印出，至於其他不能夠被顯示的字則用空白表示，但是也會將相對應的十進位值印出。

```
            !
        *********
        *       *
        *設定工作*
        *       *
        *********
            !
            +<-------+--+
            !        !  !
          *****      !  !
         *     *     !  !
        * 打入字 * 不是!  !
       *  了嗎?   *---+  !
        *   ?   *        !
         *     *         !
          *****          !
            ! 是         !
            !            !
          *****          !
         *     *  是     !
        * X= 27 *--->    !
         *  ?  *  結束!  !
          *****          !
            ! 不是       !
            !            !
        *********        !
        *       *        !
        * 處理  *        !
        *  及   *------->+
        * 印出  *
        *       *
        *********
```

圖3-17　測試鍵盤的邏輯

這個程式也可以當作一個副程式使用，用它來測試是否在鍵盤上輸入了一個字；只要按下了一個鍵，您就可以測試這個鍵是否爲正確的回應，或是依照某一個鍵的輸入而做適當的動作了。圖3－18就是這個程式，您可以依您自己的想法設計出其他的解法。

```
10 TEXT: HOME: CLEAR: A = FRE(0)
90 VTAB 22: PRINT "TYPE ESC KEY TO QUIT PROGRAM."
100 X = 0:T = 0:C = 1
110 POKE -16368,0
120 X = PEEK( -16384): IF X < 127 THEN 120
130 POKE -16368,0
140 X = X - 128
150 IF X = 27 THEN 210
160 VTAB C: HTAB 10 * T + 1:PRINT X;" = "; CHR$ (X)
170 C = C + 1
180 IF C = 20 THEN T = T + 1: C = 1
190 IF T > 3 THEN T = 0
200 GOTO 120
210 VTAB 22: CALL -958: PRINT "I QUIT!"
220 END
```

圖3-18 測試鍵盤的程式

下面就是對這個程式的描述：

行號 10 一這一行將機器清除乾淨。參看 6.1 節。

　　TEXT一使電腦進入文字資料（ text ）顯示狀況。

　　HOME 一將螢幕清除並把游標（ cursor ）放在左上角的位置。

　　CLEAR 一將所有的變數及先入後出暫存器（ stack ）清除。

　　A = FRE(0) 一將所有字串（ string ）所佔的位置清除。

行號 90 一將引號內的訊息印出。

行號 100 一設定變數的初值。

行號 110 一將鍵盤的輸出綫設爲起始狀況。

行號 120 一整個程式的關鍵指令。

　　X = PEEK(-16384) 一將記憶體位置－ 16384 或 49152內的值放入X 變數內。

　　IF... 一測試X 的值是否小於 127，如果測試結果是眞則表示尚未由鍵盤輸入資料；如果X 值是大於或等於 127，則表示已經由鍵盤輸入資料了，而 X 內存的就是打入字的相對應數值。

行號 130 一將鍵盤的輸出綫設爲起始狀況。

行號 140 一將X 變數內值的高次位數元（ bit ）去掉；這個工作使得這

個值是在適當的字（ character ）的範圍之內。

行號 150 —測試打入的字是否爲ESC字；十進位的 27 = ESC 。

行號 160 —將結果顯示在螢光幕上。

VTAB C—垂直的依照C的值移動游標。

HTAB 10 * T + 1 —水平的依照 10*T+1 的值移動游標。

PRINT —印出X的十進位值及相對應的字。

行號 170 —增加C值。C內存的是 VTAB 的計數值。

行號 180 —測試是否達到了顯示幕底行（ bottom line ）的位置；如果達到了 20，那麼螢幕上的水平位置就被增加，而垂直位置則重設爲1 。

行號 190 —如果行的數目超過了3行，那麼就將行數重新設爲1 ，使得螢幕的顯示從頭開始。

行號 200 —回到程式重新得到新輸入字的地方。

行號 210 —將游標垂直地移到 22行的位置，並將其後的螢幕清除，同時通知操作員這個程式已經結束了。

這個測試鍵盤的程式將輸入字的值同時也儲存起來，並且測試是否您輸入了 ESC 字（表示放棄輸入的工作），如果您輸入的字不是 ESC ，那麼電腦就將您輸入的字顯示出來。這個程式中行號120的程式行形成了一個廻圈，一直要到您輸入了一個字時才能夠離開這個廻圈。

您可以在這個程式中加入某些程式行，使得您可以顯示出控制字（ control characters ）或是黑白相反的字（inverse character ）。

3.3　輸入／輸出的格式

在第2章中我們大略地討論過了如何使輸入／輸出的格式令人愉悅的方法；在這一節中我們的討論將考慮到使用外圍設備的觀點；這對您來說是非常重要的，因爲當您學習到了某一個程度時，您可能就需要使用印表機之類的設備印出您所要的結果了。

3.3.1 控制印出的位置

水平的控制印出的位置是可以用許多種不同的方法達成的。

第一種方法是最顯而易見的，那就是您使用 HTAB 指令的時候；例如，HTAB 15 會將游標放在螢幕的第 15 個位置上，這個指令使您能夠將游標放置於任何一行的任何一個水平位置上。對於這個指令的一個有趣的特點是它能夠在任何行上任意移動而不會將原先輸出的訊息遮蓋，除非您在移動的同時也輸出資料才會將原先位置上的訊息掩蓋住。

第二種方法是使用 PRINT TAB(BASIC 中提供的功能)指令；在這種情況下游標將依照您提供的數字由某一行的最左邊移動到您指定的位置，或是下面的顯示行的位置，如果您在一個指令中使用不只一個的 TAB ，那麼輸出的情形仍然按照正確的方式執行，只要您所指定的位置是在目前游標所在的位置之後就可以了。但是，如果游標的位置是在您所指定的位置之後的話，那麼產生的結果就會被放在下一個機器設定的顯示位置上。

第三種在一行上設定游標位置的方法是使用 POKE 36,pos 位置的能力；在這種情況下您可以確實地將游標置於您所要求的位置上。這是一個非常方便的方法，對於輸出到印表機上的功能尤其顯得重要；詳細的情形我們將在下一節討論。

最後一種控制位置的方法就是使用 SPC( ) 的功能，這個指令使得游標依照括弧內的值往前跳過指定數目的空白；任何相關的移動都可以使您輕易地跳過螢幕或印表機上的位置。

到目前爲止我們所討論的都是水平移動的方法，當然您也可以使用VTAB指令在螢幕上垂直地移動，它與HTAB 執行的方式相同；您可以試著設計一個程式測試一下這些控制移動位置的功能。

最後您還可以設定螢幕往上捲的能力及範圍，這是由控制記憶體位置32, 33, 34 及 35 內的值而達成的。請參看附錄C 中的 POKE 命令。

我們可以使用三種監督程式(monitor)的副程式將螢光幕清除乾淨，它們也是在附錄C 中描述給您；至於這三個副程式我們使用的方法分別是 CALL -936 ， CALL -958 及 CALL -868 。

現在您已經有了控制螢光幕的各種工具了，您可以試著使用這些方法建立您自己的程式。

3.3.2 格式的控制

當您想要控制輸出到印表機的格式，您首先就要使印表機能夠接受您的命令，然後您才能將輸出的資料印在紙上。圖 3 － 19就是使用軟體控制的方法，在目前我們還不會對這個程式做詳細的討論，因爲在 3.4 節會對它做說明的，目前您只要接受它就好了。

```
串列介面                平行介面
------                  --------
100  PR #1              100  PR #1
                        110  PRINT CHR$(9);"80N"
     :                       :
     :                       :

                        500  PRINT CHR$(9);"40N"
510  PR #0              510  PR #0
```

圖3-19　串列及平行介面的程式段

這個程式設計的基本假設是印表機的介面卡(interface)與Apple 主機上的第 1 個擴充接點（ slot ）連接的。

現在我們就可以討論如何控制輸出到印表機上的動作了；如果您想將位於某兩個行號之間的訊息印到紙上，例如行號 100 到行號 500。這些行號只是一個例子而已，在實際使用時，您可以使用任何的組合。您只需用LIST 命令就可以了。

當您想在螢光幕上空一行的位置，那麼您在 PRINT 指令中就不要加入任何元素；但是在輸出到印表機上而想空一行的位置時，您就必須使用“PRINT”指令了。

如果您想將不同的資料印在同一行上，您可以寫出下面的敍述；

```
1000 PRINT A$;: POKE 36,22: PRINT B$
```

這個敍述包含了下面的意思：

* 一將 A$ 內的資料印出
* 一不做跳行的動作
* 一將游標移到 22 的位置
* 一將 B$ 內的資料印出
* 一執行跳行及游標回頭（carriage return）的動作

請您參考一下 PRINT 敍述的各種型式；注意當您將 22 這個值放入記憶體位置 36 之內時，會使游標被放置在輸出緩衝區（ buffer ）的第23個絕對位置上，B$ 將被由這個位置開始印出到紙上，就如同顯示在螢光幕上一般。

TAB 及 SPC 同樣的在印表機上發生與螢幕上相同的效果。

注意：如果您的印表機在不能夠回頭的話，那麼您在輸出訊息時必須依照順序印出您的資料，否則您可能發生很大的麻煩。

3.3.3 大／小寫字體

如果不在Apple 的硬體上做一些修改或是使用高解析度圖彩（high-resolution graphics ）的一些功能，而想要使Apple][的螢光幕上出現小寫字體那簡直是不可能的。

然而，我們在印表機上利用適當的軟體印出小寫字體卻是一件可以達成的事；一般我們都可以使用像是APPLEWRITER之類的 Word Processor 程式使小寫的字能夠印到紙上；事實上這本書中的許多英文字母都是使用 APPLEWRITER 印出來的結果；當然我們使用的印表機要能夠印出小寫字體才可以。

3.4　外圍設備的考慮

這一節討論的是使用外圍設備（ peripheral devices ）時必須從事的軟體及硬體的設定工作。對於系統的能力與適用性之間的關係我們都用實際的例子做說明。

介面裝置就是電腦與外界設備聯繫的主要工具，它使得兩者形成一個完整的操作單位。

在Apple][上有四種介面裝置提供給您使用；第一個就是遊戲控制器的接點，第二個則是螢幕的接頭，第三就是磁帶機的接點以及第四個的發聲裝置。這四種介面都是設定在機器本身的，不需要任何額外的硬體設備。

我們在本節中所要討論的是插入擴充接點上的介面卡，它們都需要額外的硬體設備才能夠使外圍的輔助器材適當的工作。

3.4.1　APPLE II的BUS

Apple][並不像LSI-11及 S-100 本身具有完全功能的bus；事實上，它也並不需要這樣的bus，因為像是鍵盤、記憶體、發聲及影像的產生器都已經與主機板連接在一起工作了。這個機器提供的是八個具有 50 個 pin 的連接點（擴充接點）供磁碟機、印表機、繪圖板（ graphics tablets ）等等與主機連接。

與任何一個接點做聯繫的工作事實上是非常簡單的；Bus傳送資料是以 16 個數元位置及8個數元資料的方式達成的，這種型式在微電腦來講是非常普通，所有的介面卡都是利用這種方式與bus做聯繫。

3.4.2　串列介面卡（ serial interfaces ）

這一節描述的是串列介面卡，對於它的組成元素，以及使用它的軟體都有詳細的說明，同時也對訊號的頻率，檢查碼（ parity ）等等做討論的工作。

大多數的外圍設備，像是印表機，MODEMS，以及終端機都是以ASCII的型式傳送資料的，由於這些設備傳送的速度都不快，因此大都是使用串列的方式做這個工作。

對於串列方式的聯繫有兩種標準的型式，它們分別是EIA RS-232C及20ma 同步廻圈的標準型式。這兩種型式中的RS-232C 事實上是最標準的一種型式，大多數的電腦都以它做為介面。

RS-232C 的標準規定在使用資料的設備與傳送資料的設備間，資料是以串列的二進位方式傳送的。這種標準介面的電子與機械方面的構造都是廣為人知的；它使用了 25 條資料綫；大多數的終端機只需要其中的3到5條就可以與電腦作聯繫了；表3－1就是這些綫的功用及解說。

資料在電腦與電腦之間傳送或是在電腦與外圍設備之間傳送可以以同步或非同步的方式進行；非同步的傳送可以說是一種 START-STOP 的型式，它不需要由主電腦傳送計時的訊號。當電腦做非同步的傳送工作時，每次傳送一個字，這個被傳送的字是以一個 START 訊號開始的，而以STOP訊號結束，至於在 START 及STOP之間的訊號就是這個字。

表3－1　RS-232C訊號

PIN	描　述
1	接　地
2	傳送資料
3	接收資料
4	要求送資料
5	送資料前的清除工作
6	準備接受資料
7	接地訊號
8	測試接收訊號綫
9	測試資料
10	測試資料
11	不被使用

12	第二條訊號測試綫
13	第二條清除資料綫
14	第二條傳送資料綫
15	DCE
16	第二條接收資料綫
17	DCE
18	不被使用
19	第二條要求送資料綫
20	終端機資料
21	訊號品質測試綫
22	指示器
23	DTE/DCE
24	DTE
25	不被使用

"5"依照ASCⅡ被傳送

```
    Stt 0  1  2  3  4  5  6  7 St St
---+  +--+  +--+  +--+--+  +--+--+--+-
 M !Sp!M !Sp!M !Sp!M  M !Sp!M  M  M
   +--+  +--+  +--+     +--+

M =Mark,Sp =Space,St =Stop,Stt =Start

     轉換成： 10101101 = AD
              !  !!  !
       lsb =  <  !!  >  = msb
              !  !!  !
              +--++--+
               !   !
               5   3    = 相反的順序排列
```

圖3-20　典型傳送字的型式

每一個訊號就是一個數元；圖3－20就是一個典型的傳送字的型式，它使用了一個 START 數元，兩個STOP 數元以及8個資料數元傳送一個字的資料。

您在圖中應該注意到了在每個字之間訊號都是處於“ 1 ”或“ 高 ”的狀況下，這就表示如果在傳送的過程中有任何的延遲等等的狀況發生時，傳送的訊號綫將不會傳送不必要的訊號，而接收的部份則等待下一個字或是這個資料的傳送完畢。

同時我們並將“ 5 ”這個字的 ASCII 字依照相反的順序傳送；因此，您所見到的數元的型式就是以相反的方式排列的；第 7 個數元（也就是 mbs(most sigiuficont bix) 也被設定爲 1 表示使用的是偶數的檢查碼（ **even parity** ）。

現在我們要定義一些名詞使您對於串列的傳遞訊息能夠更進一步的瞭解：

傳遞速度（ Baud ）一定義爲在某一段時間內傳送資料的數量；一般以數元 / 秒（ bps ）爲單位。

資料的長度一指的是供輸出字佔用的緩衝區（ buffer ）的大小。

檢查數元一分爲偶數及奇數的檢查數元，也就是使數元組內數元值爲 1 的數目爲偶數或奇數的數元。

檢查總和（ Checksum ）一在一個區（ block ）內所有字的總和存在這個數元組內。

START 數元一指的是每一個字的開始數元；一般有一個。

STOP 數元一指的是每一個字的結束數元；一般有兩個。

資料數元一指的是每一個字傳送的數元數目；一般是 8 個。

Apple 的串列介面卡具有十個可以由使用者訂定的元素，它們有的需要由您設定，有的則設爲指定值。其中有四個元素當這個卡片被插入電腦內時就已經被設定了；它們分別是：

```
元素         開關               指述
----------  --------          ------------
1-傳送速度   1    2    3
             -----------
             on   on   on        110    baud
             off  on   on        134.5  baud
             on   off  on        300    baud
          ** off  off  on       1200    baud
             on   on   off      2400    baud
             off  on   off      4800    baud
             on   off  off      9600    baud
             off  off  off    19,200    baud

2- 游標回頭       4
   的延遲         -
                  on          不可以
               ** off          可以

3- 行的寬度       5    6
                  -------
                  on   on    40 char/line
                  off  on    72 char/line
               ** on   off   80 char/line
                  off  off  132 char/line

4- 跳　行         7
                  -
               ** on          不可以
                  off          可以
```

永久設定NT DEFAULTS

1- 不理會偵錯數元
2- 不理會檢查總和
3- 小寫轉換成大寫
4- 資料數元數 = 8
5- Start 數元數 = 1
6- Stop 數元數 = 2

注意： **被設定的開關

圖3-21　Dip開關的設定

現在我們已經知道了串列介面卡如何傳送資料了，目前我們所要討論的則是如何使用 APPLESOFT BASIC 控制串列的訊號交換。當我們將串列介面卡插入電腦的一個擴充接點上時。（一般我們將串列介面卡插在第1個擴充接點上），同時我們也假設所有的開關都已經依照前面的描述設定好了。

最後就是我們所要做的工作。——設計程式使串列介面卡工作並將結果印出到印表機上。

```
不是-DOS                 DOS
    :                     :
                     100 D$ = CHR$(4)
110 PR #1            110 PRINT D$;"PR#1"
    :                     :

     （進行輸出的工作）
    :                     :

220 PR #0            220 PRINT D$;"PR#0"
    :                     :
```

行號110的敍述指定串列介面卡位於第1個擴充接點；行號220的敍述則使輸出由第1個擴充接點返回到螢光幕去。

3.4.3 平行介面卡（Parallel Interface）

我們在這裏討論平行介面卡並且與串列介面卡相比較，同時我們也把使用這個卡片的軟體介紹給您。

與具有平行介面卡的外圍設備做訊號交換的工作有幾點主要的不同；平行的聯繫是由電腦將所有的數元同時的輸送到外圍設備內；也就是說數元0到7同時的達到印表機；因此，與電腦及外圍設備連接在一起的帶子（cable）就必須具有八條資料綫及控制綫，這條控制綫就形成了控制資料輸送次序的“ handshaking ”綫。圖3－22就是控制的軟體。

```
    :
110 PR #1
120 PRINT CHR$(9);"80N": REM set 80 cols
    :

       (進行輸出工作的程式段)
    :

220 PRINT CHR$(9);"40N": REM set 40 cols
230 PR #0
    :
```

圖3-22 平行輸出印表機的程式段

3.4.4 訊號交換介面卡(communications interface)

訊號交換介面卡的功用是爲了要使兩部電腦之間能夠做訊號交換的工作，它是以串列的方式傳送資料的。一般而言它的傳送速度是110或300，(300是設定值)。

下面有三種不同的連繫形式，它們分別是：

* 一單向式(**Simplex**)
* 一半雙向式(**Half Duplex**)
* 一雙向式(**Duplex**)

單向式的傳送訊息是以一個方向的方式執行的；半雙向式則在某一個時間是以一個方向而在另一個時間又可以以另外一個方向傳送；至於雙向式則可以同時以兩個方向傳送資料。您可以由鍵盤控制某一種傳送資料的方式；在半雙向式的情況下，您可以打入CTRL A CTRL H達成目的；使用CTRL A CTRL F則可以達成雙向式的傳送方式；至於傳送速度的改變則可以使用CTRL A CTRL 1使得速度由300變爲110，而後又可以使用CTRL A CTRL 3使速度變回300。

如果您想使用訊號交換介面卡控制印表機，那麼您只需將一個控制印表機的程式存入記憶體內，CALL 它那麼您就有一片串列印表機介面卡了（以 300 bps 的速度工作）。

至於詳細的使用這片介面卡的情形請參看訊號交換介面卡的手册。

3.5 總　結

這一章中討論的是關於程式的組織，完整性、速度、主程式、副程式等等的問題，討論的方向在於如何提高程式的可靠性、完整性及加快執行的速度。

我們花了許多的時間討論副程式及 FOR...NEXT 的指令，原因是它們非常的重要，您將在以後常常的用到這些結構，同時它們對您的幫助將是非常大的。

另外在本章中一個主要的部份就是第 3.2 節，在這一節中我們討論的是程式的結構，由於這些討論使得我們在設計程式時能夠設計出更完善、可靠度更高的程式；我們相信即使您是一個不限格式的程式設計師，在您研究過了這些結構的討論之後，相信對您設計程式的幫助仍然是非常大的。

至於本章的最後四節討論的是外圍設備的介面卡， Apple 的串列，平行及訊號交換介面卡都已經討論過了，這些討論相信已經使您的程式能夠與外界的設備做交換資料的工作。

在您讀完本章之後，您已經可以設計出相當完整、速度快及結構完善的程式了，同時您也可以控制您的輸入與輸出設備，並將輸入與輸出資料的格式定得令人滿意。

第4章

程式的結構

4.0 大 綱

這一章我們將討論程式的結構問題，討論的重點放在記憶體的使用，變數的組織上面，同時也對“手册”(menus)及軟體應用程式作一個介紹；最後要討論的則是一種程式的架構，在使用這種架構時，您可以將許多的應用程式結合在一起供以後使用。

第 4.1 , 4.2 及 4.3節描述的是 Apple][記憶體的使用情形以及變數(variable)及字串(string)表的組織情形。

程式是人們為了要使電腦工作而設計的，同時也希望能夠令其他的人輕易的瞭解；程式並不是為娛樂而存在的，而是為了決定什麼要被做及什麼將被做。

確實地瞭解程式的目的對我們來說是絕對必要的，一個程式並不是只要能對測試值做正確地反應就可以了，它必須對所有的資料都能產生正確的結果。

如果想要瞭解整個程式，那麼我們必須對程式語言相當的熟悉，而對程式設計師來說他(或她)的責任就是要把有組織及結構完善的程式設計出來，使得往後研究這個程式的人們能夠很快地瞭解它的意義。

4.1 記憶體的使用

爲了要使您能夠瞭解記憶體的使用情形，我們現在要設計兩個小的程式幫助您瞭解我們討論的細節以及資料儲存的方式。這兩個程式我們分別稱它爲 TEST #1 及 TEST #2 。

我們相信這兩個程式並不需要逐項地討論您就可以瞭解了，但是我們仍然在必要的時候做詳細地說明使您能夠加深印象。圖 4 － 1 及 4 － 2 就是這兩個程式。

這些程式設計的目的是要使我們能夠認識記憶體的位置，因而能夠檢查其中所儲存的資料；我們期望經過這兩個程式的示範之後，您能夠更加瞭解記憶體的格式及內容，並對APPLESOFT][及我們的電腦更多一層的認識。

事實上，我們將扮演控制記憶體的角色，至少在變數的儲存位置上；在正常的情況下，APPLESOFT][及整數BASIC會做這個工作，但是如果使用 LOMEM: 及 HIMEM: 指令，您就可以控制您自己的記憶體了。

位於行號 100 到 120 的程式行（ TEST #1 ）及行號 20 到 30的程式行（ TEST #2 ）中的磁碟命令（ disk commands ）將在第八章中討論。

```
10 LOMEM: 24576
15 DIM X(2),DT$(2)
20 HOME: TEXT
30 VTAB 4: INPUT "ENTER X VALUE  = ";X(1)
40 VTAB 6: INPUT "ENTER Y VALUE  = ";X(2)
50 VTAB 8: INPUT "ENTER 1ST DATE = ";DT$(1)
60 VTAB 10 :INPUT "ENTER 2ND DATE = ";DT$(2)
70 PRINT : PRINT X(1);"  ";X(2);"  DATE
   S = ";DT$(1);"  ";DT$(2)
100 PRINT CHR$(4);"BSAVE VTBF,A$6000,L$200,D1"
110 PRINT CHR$(4);"BSAVE STBF,A$95F4,L100,D1"
120 PRINT CHR$(4);"RUN TEST #2,D1"
130 END
```

圖4-1 Test# 1 程式

在這些程式被輸入記憶體內之後，就必須以 TEST #1 及 TEST #2 的名稱儲存到磁片上去；至於在磁碟命令中的 $ 是表示十六進位（ hex ）數字

```
10 LOMEM: 24576
15 DIM X(2),DT$(2)
20 PRINT CHR$(4);"BLOAD VTBF,A$6000,D1"
30 PRINT CHR$(4);"BLOAD STBF,A$95F4,D1"
40 PRINT
50 PRINT X(1);"  ";X(2);"  DATES = ";DT$(1);"
";DT$(2)
60 END
```

圖4-2 Test# 2程式

。請參看第八章及附錄D。

TEST #1 做以下的工作：

行號 10 ：這個命令將記憶體最低位置的指標設為 24576（ HEX 值的 $6000 ）。這麼做的原因是要能夠輕易地發現變數表（ variable table ）的起始位置；正常的情況下，變數表是被存在APPLESOFT][程式之後的，如果我們允許這種情形發生，那麼我們就必須計算變數表的位置了（我們可以使用兩個 peek 指令達到目的）；但是我們希望它位於我們指定的位置內。

行號 15 ：設定兩個向量的行列－X 及 DT$，用來儲存輸入的變數值。

行號 20 ：清除螢光幕並設定文字資料（ text ）狀況。

行號 30 ：接受 X 值並存在 X(1) 內。

行號 40 ：接受 Y 值並存在 X(2) 內。

行號 50 ：接受第 1 個時間值並存在 DT$(1) 內。

行號 60 ：接受第 2 個時間值並存在 DT$(2) 內。

行號 100 ：存入變數表， VTBF 。

行號 110 ：存入字串表， STBF 。

行號 120 ：執行 TEST #2 程式。

行號 130 ：結束這個程式。

當TEST #1 被執行之後，程式將依次要求您輸入X，Y， 1ST DATE及

2ND DATE（這些日期只是供做字串之用，並不代表任何意義），輸入的資料將被顯示在螢光幕上，接著磁碟機就開始轉動並且將 VTBF 及 STBF 這些二進位檔（binary files）存入磁片內。最後的一個步驟則是由 TEST # 去執行 TEST #2 程式。

TEST #2 做下面的工作：

行號 10：設定記憶體的最低位置，以便用來儲存變數。

行號 15：與 TEST #2 般的設定 X 及 DT$ 向量的行列。

行號 20：將 VTBF 這個二進位檔存入記憶體內。

行號 30：將 STBF 這個二進位檔存入記憶體內。

行號 40：在螢光幕上顯示一行空白行。

行號 50：顯示出 X ，Y 第 1 個日期及第 2 個日期的值。

行號 60：結束這個程式。

TEST #2 程式不需要操作員做任何的動作，它的主要工作就是將 VTB 及 STBF 這兩個檔存入記憶體指定的位置內，再將由 TEST #1 輸入檔內的資料顯示出來。

現在我們已經知道了 TEST #1 及 TEST #2 的功用了，我們把注意力由這兩個程式轉移到它們在記憶體內的什麼位置發生這些動作的問題上去。爲了要達到這個目的，我們必須進入監督系統（monitor）並且審查適當的記憶體位置看看其中存的是什麼資料。

```
]RUN TEST #1,D1

ENTER X VALUE   = 123

ENTER Y VALUE   = 246

ENTER 1ST DATE = 041880

ENTER 2ND DATE = 041781

123   246   DATES = 041880   041781

123   246   DATES = 041880   041881
```

進入監督系統的方法就是呼叫它；這是在立即式執行的情況下輸入CALL -151 並按下RETURN鍵而達成的。

```
*800.8FF

0800- 00 0C 08 0A 00 A4 32 34   ......24
0808- 35 37 36 00 1D 08 0F 00   576.....
0810- 86 58 28 32 29 2C 44 54   .X(2),DT
0818- 24 28 32 29 00 25 08 14   $(2).%..
0820- 00 97 3A 89 00 41 08 1E   ..:..A..
0828- 00 A2 28 34 29 3A 84 22   ..(4):."
0830- 45 4E 54 45 52 20 58 20   ENTER X
0838- 3D 20 22 3B 58 28 31 29   = ";X(1)
0840- 00 5D 08 28 00 A2 28 36   ...(..(6
0848- 29 3A 84 22 45 4E 54 45   ):."ENTE
0850- 52 20 59 20 3D 20 22 3B   R Y = ";
0858- 58 28 32 29 00 82 08 32   X(2)...2
0860- 00 A2 28 38 29 3A 84 22   ..(8):."
0868- 45 4E 54 45 52 20 31 53   ENTER 1S
0870- 54 20 44 41 54 45 20 3D   T DATE =
0878- 20 22 3B 44 54 24 28 31    ";DT$(1
0880- 29 00 A8 08 3C 00 A2 28   )...<..(
0888- ...........
```

圖4-4　Test＃1的解碼情形

我們在這些機器語言的符號右邊加上了適當的說明，使得您能夠清晰地瞭解它們的意義。在圖4－5中的圖就是變數表，圖4－6中的則是字串表。

```
*6000.6028

6000- 58 00 16 00 01 00 03 00   X.......
6008- 00 00 00 00 87 76 00 00   ........
6010- 00 89 64 00 00 00 44 D4   ......D.
6018- 10 00 01 00 03 00 00 00   ........
6020- 06 FA 95 06 F4 95 00 00   ........
6028- FF
```

圖4-5　Test＃1的變數表

```
*95F0.9607

95F0- 00 00 00 04 30 34 31 37  ....0417
95F8- 38 31 30 34 31 38 38 30  81041880
9600- F4 00 00 00 00 00 00 00  ........
```

圖4-6 Test# 1 的字串表

在繼續討論之前，我們先定義幾個名詞；它們分別是：

lsb—最低位的數元（ Least significant bit ）

msb—最高位的數元（ Most significant bit ）

LSB—最低位的數元組（ Least Significant Byte ）

MSB—最高位的數元組（ Most Significant Byte ）

```
   76543210   <--數元位置
   !      !
 0 01010110   <--二進位型式的數元組資料
   !      !
 msb      lsb
   !      !
   +--++--+
    !   !
    5   6     <--十六進位的數元組資料
    !   !
    +---+
    數元組
```

數元位置零（ 0 ）表示 2 ∧ 0 = 1 ，而數元位置 7 則表示 2 ∧ 7 = 128 。

在這兩個程式中極爲有趣的一件事，就是您可以在兩個程式中傳送變數值而不需要由鍵盤做輸入的工作或是使用文字資料檔。這是由於我們在第 1 個程式中將 VTBF 及 STBF 存入磁片內的原因；只要您在程式中使用 DIM 敍述小心地安排同樣次序，大小及位置的行列式，您就可以利用DOS中的二進位檔的能力（參看第八章）重新將值存入正確的記憶體位置內了。

您將發現這個小小的技巧對於某些狀況是非常有用的。同時我們可以發現 TEST #2 並不需要加入也可以查看記憶體內的資料，但是我們這麼做只

是要使您對程式的能力更加一層瞭解。

4.2　變數表

由APPLESOFT][處理的變數表是用來儲存變數值（數值的及字串的）；在APPLESOFT][中，變數表是被存在 BASIC 程式的後面的，當然，我們也可以使用 LOMEM 命令重新設定變數表的儲存位置，這個表的組織情形就如同圖 4 － 5 一般。

前面的兩個位置－ $6000 , $6001 ，用來儲存佔用兩個字的變數名稱；在上面的情形下， ASCII 58 就表示X 。

接下來的兩個位置－ $6002 ， $6003表示的是這個變數佔用了多少個數元組；在上面的情形下，就是十進位的 22 或十六進位的 16 。

接下來的兩個數元組－$6004 ，$6005則表示X 變數是一個一階的向量。

再接下來的數元組－ $6006則告訴我們在這個向量中共有三個元素，也就是 0 ，1 及 2 。

現在接下去的五個數元組－ $6007 到 $600B 存的就是第 0 個元素的值；裏面存的就是 0 。至於第 1 個元素內的值是 123 就存在 $600C 到 $6010 的記憶體位置內，而 $6011 到 $6015 內存的則是第 2 個元素內的值 246 。

相信您在看了圖 4 － 5 之後，立刻想問的問題就是“您如何由 877600-0000 得到 123 這個值？”；這是一個適當的問題，因此我們現在就解釋給您。首先，我們必須知道，我們由鍵盤輸入的值是用十六進位的方式在記憶體內表示十進位值的；接著，我們要解釋的是如何使用 5 個數元組表示一個元素。然而，在我們做這個工作之前，您必須瞭解浮點數字（ floating point numbers ）在記憶體內的安排情形。每一個浮點數字都是由兩部份組成的。它們是：

指數・數值

指數佔用了五個數元組中的第 1 個數元組，指數是以超過 $80 的正負數字表示的－這表示它是使用 $80 做爲它的零值。

十進位的小數點（在這種情況下就是二進位的小數點）被設定在msb的右邊；因爲在二進位的浮點數字的表示中，msb永遠是1；這個數字的符號（＋＝0，－＝1）被存在這個位置，但是當做算術運算時，這個數字的符號會被去掉並存在其他的地方。

數值部份佔用了剩下的4個數元組；因此一個浮點數字就依照下面的方式儲存在記憶體內。

```
EXP  HOB  MOH  MOL  LOB
 !    !    !    !    !
 !    !    !    !    > = 低次位數元組
 !    !    !    !
 !    !    !    > = 中間較低次位的數元組
 !    !    !
 !    !    > = 中間較高次位的數元組
 !    !
 !    > = 高次位數元組
 !
 > = 指數
```

```
   7     B     十六進位    = 123  十進位
 +--+  +--+
 !  !  !  !
0 0111 1011      二進位    = 123  十進位

123  十進位  = 7B hex = 01111011  二進位
```

告訴您以上情形的目的是要使您能夠瞭解資料是以什麼形式以及儲存在什麼地方的問題；在正常的情況下，您根本不必去考慮它是如何做的。

4.3 字串表

字串也是被存在變數表內的，但是字串值是由 HIMEM 位置開始儲存的。字串表往下繼續存字串資料，朝的是變數表的位置。

請參考圖4－5中的$6016到$6025的位置。

前面的兩個數元組—$6016及$6017存的是變數名稱DT$，而$6017的高次數元則被設爲1，這使得數值與字串值能夠分辨清楚。

數元組 $6018 及 $6019 表示 DT$ 變數佔用了16 個數元組，也就是十六進位的10 。

$601A 數元組內存的就是表示 DT$ 是一個一階的行列。

$601B － $601C 表示DT$有三個元素。

$601D － $601F 是用來儲存元素 0 的資料的，目前裏面是空的。

$6020 － $6021 數元組內存的是6 ，就是元素 1 的長度及這個字串開始儲存的位置 $95FA 。

第 2 個元素則依照同樣的方式安排，這個元素的長度也是 6 ，並且由 $95F4 開始儲存。

如果您現在參看圖 4 － 6 ，您可以發現第一個日期＝ 303431383830 ，並由 $95FA位置開始儲存，上面的日期可以轉換成 041880 。

您必須注意的是第 2 個日期儲存的位置較第 1 個爲低，這就表示字串表是往下儲存的（往變數表的方向）。

4.4　程式的編譯

一個 BASIC 程式被存在記憶體內，當要執行它時，APPLESOFT][語言必須能夠將每一個指令編譯或適當的型式使這個程式能夠被正確地執行。

正常的情況下，一個APPLESOFT][的程式是由記憶體位置 2048＝ $800 的地方開始儲存的；參看圖 4 － 4 的 TEST #1 程式就可以發現這個情形；至於整數 BASIC 的程式則必須由 HIMEM 開始往下存。

第 1 個數元組（ $800 ）存的是一個零，一個零值的數元組被用來做爲程式中指令間的分隔。

接下去的兩個數元組依照 LSB MSB 的順序存的是下一個指令在記憶體中的位置。

再下去的兩個數元組內存的則是這個指令的行號。

至於剩下的數元組（直到下一個指令開始的位置）存的就是這個指令本身。

我們對 TEST #1 程式的一部份做了翻譯的工作，這使得您能夠瞭解這

個程式，也有一部份我們沒有做翻譯，那些數元組內存的是無法顯示的 ASCII 字。然而，所有的指令並沒有失掉，您可以參看附錄F，您將可以翻譯這個程式的其他部份。

最後一個必須解釋的項目就是位置的結構。$801－$802數元組內存的是下一個指令的位置，而這個位置是以下面的格式表示的；

LSB MSB = 0C 08 = $080C hex = 2060 十進位

這個位置能夠在分隔每一個指令的空白的後面看到；注意的是位置是依 LSB MSB 的格式排列的。

```
00  0C  08  0A  00  A4  32  34  35  37  36  00
!   !    !  !    !  !   !               !   !
!   +--+--+ +--+--+ !   +-----+---------+   !
!      !       !    !         !             !
!      !       !    !         > = 24576     !
!      !       !    !                       !
!      !       !    > = LOMEM:              !
!      !       !                            !
!      !       > = 10                       !
!      !                                    !
!      > = $080C = 下一個指令的位置          !
!                                           !
> = 分  界          ------------------------+
```

每一個指令都可以依照上面的方式分隔出來並加以翻譯；注意的是在指令中間並沒有任何的空白存在，所有指令中間的空白都是在您 LIST 程式時才加入的。

在我們輸入程式之後，第一件我們想要做的事就是去執行它；首先，我們急於看看我們的作品，其次則是我們希望看到我們的程式在產生下列訊息之前能夠執行多久。

SYNTAX ERROR

BREAK IN 1

任何輸入的程式在被執行之前都必須先被轉換成某種機器語言（machine language）的格式；下面有三種做這個工作的方法；但是對於任何一種語

言來說，只能夠使用其中的一個方法；

* 一編譯程式（Compiler）
* 一間接編譯程式（ Incremental compiler ）
* 一編譯器（Interpreter ）

讓我們由最後一個項目開始，編譯器具有立即編譯打入的命令或指令的能力，同時找出可能的錯誤，如果您輸入的是合乎語法的指令，那麼您就可以輸入下一個指令。

然而，如果您的指令中存有語法的錯誤，那麼您就會在螢光幕上看到一個錯誤的訊息，而這個指令就不會輸入到程式中去，您可以重新的輸入它或是根本不要這個指令。整數 BASIC 就是以這種方式作業的。

接著在執行這個程式時（ RUN ），每一個指令在處理的過程中都順序地被編譯成機器語言而執行。

至於編譯程式作業的情形就完全不同了；在這種情況下您經由 EDITOR 輸入整個的程式；EDITOR 的功能就是接受您輸入的每個指令並將它存在某個地方（磁碟內或記憶體內），然後繼續這個動作直到您告訴它輸入完畢了爲止。接著您才要求將輸入的程式編譯，這表示所有輸入的 BASIC 敍述都經過分析變成機器語言碼，而這些機器語言碼就存入一個檔（ file ）內（這個檔被稱爲原始檔（object file ）。您現在已經有了兩種型式的檔了，一個是您原先設計的程式（ source code ），一個則是由編譯程式產生的原始檔；至於編譯的過程中如果編譯程式發現您的程式中有任何錯誤，那麼它將產生錯誤訊息，當產生錯誤時，您將被要求修正這些錯誤，然後編譯才能繼續。

現在當您執行這個程式時，事實上您是在執行這個沒有錯誤的原始檔，但這並不表示程式中所有的錯誤都已經不存在了，這只是表示編譯程式到目前爲止還沒有發現其他的錯誤罷了，像邏輯的錯誤仍然存在。如果在執行中發現任何錯誤，那麼您必須回到您輸入的程式去修正它，重新的編譯過，然後您才能再執行您的程式。

間接編譯程式是介於編譯器與編譯程式中間的一種編譯工具；它將可以

翻譯的程式段編譯成機器碼，然後將其他的部份留給編譯器編譯。

APPLESOFT][並不是一個完全的編譯器，因為它並不會給您輸入時的錯誤訊息，它只是給您執行時的錯誤訊息。它在執行程式時立即編譯機器碼，並在遇見第一個錯誤時停止程式的執行，並將錯誤訊息顯示出來。

4.5 手　冊（MENU）

當一個程式設計師想要將設計的程式供應市場的需求時，一個主要的考慮就是必須讓使用者能夠依照程式的需求作正確的反應；同時，也必須讓使用者能夠容易地使用這個程式。

為了要使操作員能夠做正確的選擇，程式中就必須使用手冊（ MENU ）的格式，使得操作員能夠選擇他（或她）所需要的處理程序。下面就是一些使用手冊所能得到的好處：

* 一使操作員工作方便。
* 一使用者能夠決定選擇的項目。
* 一程式設計容易。
* 一操作員的訓練簡易。
* 一尋找錯誤的過程較簡單。
* 一容易設計出程式。

在一個具有手冊的程式中，所有控制選擇的工作都必須在主程式中完成，您可以參看圖 4 － 7 的程式。

由於手冊是這樣的重要，因此必須小心的設計，才能夠使得操作員的選擇是合法及有效的；同時必須確定操作員的意向，這樣才能在程式中做正確地反應工作。

4.6 副程式

副程式以及它的結構我們已經在第三章中討論過了，現在的主題則放在副程式的位置上面。

在一個程式中有兩個通常被使用供副程式放置的區域；它們是程式中專供副程式使用的區域（稍後討論），以及程式的主要程式段中間的一部份。

除了它們的功能及大小之外，副程式大都是相同的。我們爲了討論方便起見，將副程式分爲大型副程式（ Major blocks ），中型副程式（Minor blocks ）以及一行副程式（ Single line routines ）。

4.6.1 一行副程式

首先我們要討論的就是將程式行編入副程式的一般規則。如果程式行，就如一行的程式行，需要在一個程式中被執行兩次以上，那麼我們就可以將它編入副程式中使用。下面就是一個例子：

```
55 INPUT "WHICH ONE? ==> ";N :RETURN
```

在這種情況下我們將把這個副程式放在一般副程式的外面，請參看 4.7 節。注意這個副程式的行號很低；因此，當程式中呼叫它時，將會很快地被找到而執行；這是因爲行號的尋找方式是由起始行號開始並且按照行號的大小依序尋找的。

一行的副程式就如它的名字一般，只佔用了一個行號的程式行，但是它可以擁有許多個指令；但是在大多數的情形下，它只能產生一個功能。

4.6.2 中型副程式

假設您有一個印製報表頭的中型副程式，同時，您希望在報表的每一頁上都印出同樣的表頭；在這種情況下，這樣的一個副程式也許就佔用了 5 到 15 行的程式行。

像這樣的程式段就適合被放置在專供副程式放置的區域內；如果一個中等大小的程式段（佔用了一個數目的程式行）被使用的次數超過兩次，那麼我們就需要使用一個副程式來處理它。例如下面的程式段：

```
60 VTAB 20 : HTAB 13 : INVERSE : PRINT "ILLEGAL
    ENTRY!";CHR$(7) : NORMAL
62 FOR D = 1 TO 1000: NEXT D : RETURN
```

這個小小的副程式可以被使用來指出由鍵盤輸入的錯誤，錯誤訊息將在每次這個副程式被呼叫時顯示在螢光幕同樣的位置上。（注意，這個副程式可以達成兩個目的：第一它能夠指出錯誤並造成一個延遲（ delay ），其次如果您的程式中是 GOSUB 62 ，那麼它可以只造成一個延遲）。

4.6.3 大型副程式

第三種大小的程式段就是大型的副程式，它可能佔用三到四個螢幕的顯示面積，這種型式的程式段有一個特點，那就是只要執行進入其中，程式就必須在這個區域內花費許多的時間。

在這種情況下，我們可能就是與手册中選擇出來的主要程式段打交道。請參看4－7節程式的架構。

由於程式將花費許多的時間在這個程式段中，因此我們可能將它放置於程式中的後面部份（具有大的行號），這就表示對任何其他副程式的呼叫將是一個往較小行號去執行的動作，這也符合我們增加程式執行速度的要求。程式在執行這個程式段時將一直地呼叫及處理接近程式起頭的程式部份。例如；

```
1000 HOME : PRINT : PRINT TAB( 12) "*** SUBR #1
     ***" : PRINT
1010  :
      :

1100 GOSUB 62      <--呼叫在程式前面部份的副程式
1110  :
      :

1995 RETURN
```

4.7　程式的架構

在圖4 － 8 中所顯示的程式的架構就是一個典型的 BASIC 程式的組織情形，它具有 BASIC 語言的特性，並取得了 Apple ，及結構程式的優點

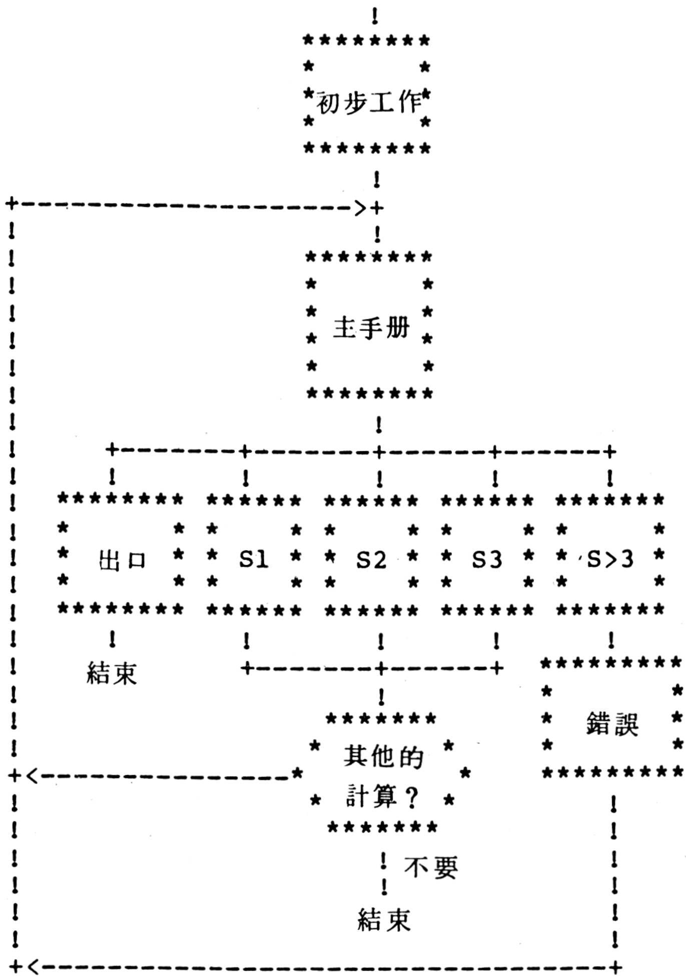


圖4-7　架構程式的邏輯圖

，上面就是這個程式的邏輯圖。

請對下面的討論參考架構的程式。

前面的九行，1 ～ 9，指出這個程式的作者，程式的名稱、日期、及使用的語言。

行號 10 ：將使用用的機器清除，參看 6.1 節。

行號 20 － 49 ：DIM 敍述，定義變數等工作。

行號 50 ：跳到主程式去。

行號 51 － 899 ：保留做爲放置經常被呼叫的副程式的位置。

行號 900 － 999：主程式的放置位置；包含了手册，手册的處理及其他的計算工作。

行號 1000 之後：保存大型副程式，程式將花費許多的時間在這一部份上。

像這樣的安排，對於一個程式而言是取得了我們到目前爲止所討論的所有優點，像是速度、結構等等。同時這個程式顯示出了結構程式的型式及構造的完整性，並且對於程式的修改及偵錯都有很大的幫助，我們覺得像這樣程式組織的情形所獲得的優點實在是非常的多。

請瞭解一點，那就是行號的設定並非一成不變的，即使我們使您覺得如此，但是事實上卻是完全由您的喜愛決定的；我們所想要表示的只是整個程式的安排情形，並且依照這些安排情形告訴您它的優點。

```
1 REM ***** SKELETON *****
2 REM *
3 REM * WRITTEN BY:JL CAMPBELL
4 REM *       DATED:MM/DD/YY
5 REM *
6 REM * LANGUAGE:APPLESOFT II
7 REM * COMPUTER:APPLE II
8 REM *
9 REM ********************
10 TEXT : CLEAR : POKE 216,0 : A = FRE(0)
20 REM * DIM STMTS & VAR. DEF.
50 GOTO 900
51 REM **************
52 REM * GEN'L SUBR *
53 REM **************
55 INPUT "WHICH ONE? ==> ";N : RETURN
60 VTAB 22 : CALL -958 : HTAB 13 : INVERSE :
    PRINT "ILLEGAL ENTRY!";CHR$(7) : NORMAL
64 FOR L = 1 TO 1000: NEXT L : RETURN
899 REM * MAIN DRIVER
900 HOME:PRINT:PRINT TAB( 9) "*** PROGRAM TITLE!
     ***":PRINT
905 PRINT "MAIN MENU:": PRINT "---- ----"
910 PRINT TAB( 10)"1- SUBR #1"
912 PRINT TAB( 10)"2- SUBR #2"
914 PRINT TAB( 10)"3- SUBR #3"
928 PRINT:PRINT TAB( 10)"0- EXIT PROGRAM"
930 VTAB 10: CALL -958: GOSUB 55
935 IF N = 0 THEN 990
940 IF N > 0 AND N < 4 THEN 950
945 GOSUB 60: GOTO 930
950 ON N GOSUB 1000,2000,3000
960 VTAB 20: CALL -958: INPUT "WANT ANOTHER
     CALCULATION?(Y/N) ";A$
962 IF LEFT$(A$,1) = "Y" THEN 900
964 IF LEFT$(A$,1) = "N" THEN 990
966 GOSUB 60: GOTO 960
990 VTAB 4: CALL -958: VTAB 10: PRINT "GOODBYE
     FOR NOW!": END
1000 HOME:PRINT:PRINT TAB( 12)"*** SUBR #1
       ***": PRINT
1500 GOSUB 64
1999 RETURN
2000 HOME:PRINT:PRINT TAB( 12)"*** SUBR #2
      ***": PRINT
2500 GOSUB 64
2999 RETURN
3000 HOME:PRINT:PRINT TAB( 12)"*** SUBR #3
      ***": PRINT
3500 GOSUB 64
3999 RETURN
```

圖4-8 架構程式

4.8 延 伸

相信您可以看出來，使用上面程式架構的方式，我們可以利用這些原則在整個程式系統（ program package ）的設計上。

使用這種方法，在程式系統中的每一個程式都可以被大型的副程式所替代，然後在主程式中的手册就可以使用情況的結構做轉移及呼叫的工作，而被呼叫的程式也就成爲程式系統中第二階層的邏輯結構了。這個被呼叫的程式同樣地也可以呼叫第三階層的其他程式；這種情形請參考圖２－８中的說明。

當一個在某一階層的程式執行完畢之後，它就呼叫在它上一層次的程式，就如同我們所描述的一般；使用這種方式的程式系統就如同程式與副程式一般地工作，而採取的方法正就是我們所推薦的由上往下的方式。

因此，我們可以知道所有結構程式的好處都可以推展到程式系統中去。

4.9 總 結

這一章討論的重點放在兩個主要的部份上。第一部份討論的是 BASIC 語言內部的結構及表示方法，以及字串與變數的保存方式和APPLESOFT][程式的內部表示方法。

第二部份討論的是程式的架構問題，我們用一個程式來代表這種情形，而我們也可以由這種情形延伸到整個的程式系統。

程式的架構與房屋的架構非常類似，您可以在這個架構上加上窗戶、牆壁等等。

至於副程式的討論則專注於放置位置的問題，我們介紹了三種不同大小的型式，每一種型式都有它存在程式中的特定位置。

最後我們所討論的就是程式的架構如何延伸到程式系統中的問題。

第5章 程式的測試

5.0 大 綱

這一章將專注於費時、費錢又必須做的程式測試的工作，這個工作可以在發展中不同的層次及不同的時間完成。

經過了許多年的時間之後，測試成為提高軟體可靠度的主要手段。最近，在大型的電腦上曾經使用了結構式的程式設計，由上往下的設計程序等等方式示範了產生更可靠軟體的方法；但是對於微電腦來說，測試仍然是最有效的手段。

一般說來，測試的目的就是消除所有程式中不需要的失敗及錯誤；為了要達成這個目的，我們設計出各種情況測試程式中的各種反應－－不論是正確的或是不需要的。尤其重要的是，我們設計出各種情況測試及練習用各種方法設計出來的程式。

我們將討論各種層次的測試，不論是單一程式行的情形或是主要程式段都將涵蓋在討論的範圍內；同時我們也介紹在程式系統中測試外圍設備的方法。

5.1 主要的程式段

在所有較複雜的程式中，必定有被認為是主要部份的程式段－－不論從大小或是邏輯的觀點來看，不論是那一種情況，這些主要的程式段都必須以

謹慎的態度，並且在發展的適當時機做測試的工作，而這些時機就是當這個程式段第一次被加入主程式中，以及當程式要被執行的時候。

第一組的測試是發生在將這個程式段加入主程式中的時候，也就是在這個程式段被連接與定標題的時候；也有可能這一個程式段的工作只是在做延遲的工作，當您進入之後就呆在這段程式裏，然後在經過了固定的時間之後返回到呼叫的程式段去。

當所有主要的程式段以及它們與主程式中間的連接都被加入並測試完成之後，您就可以輸入處理的程式行了。

在所有的處理程式行被輸入完畢，您現在就要做測試某個區內的程式行的工作了；每一個程式段都可以按照獨立的方式測試，當然測試的要點就在於語法、語意以及能否產生正確的結果等問題上，按照程式段的順序做測試的工作直到建立了完整的程式為止；然後，您就可以測試整個的程式了，這時候測試的方法就是用不同的正確資料及錯誤的資料分別測試程式的反應。

如果您增加新的程式段，而這個程式段又與主程式部份相關連，那麼您測試這個程式段的方式就與測試主程式的方式相同。

5.2 副程式段

副程式，尤其是大型的副程式可以被放置在程式中間，並且在任何的時間都可以被呼叫，檢查主程式與這個副程式之間的連接是否正確以及這個副程式能否正確地工作；請參看4.7節的程式的架構。

在對大型副程式做測試的時候所用的方法與上面所描述的測試主程式的方法相同，唯一的不同點只是您必須考慮副程式可以在不同的程式段及不同的情況下被呼叫這個問題，在這種情形時，您必須測試在每一種情況下呼叫副程式所產生的結果。

5.3 程式系統

程式系統（ Program packages ）發展與測試的方法和發展與測試一個程式的方式相同；您由一個問題開始，發展出各個部份，再測試各個部份的程式以及程式與程式之間的連接關係，最後再測試它們執行的情況，就如同它們是一個程式一樣。在這種情況下，您必須將每個程式獨立，測試每個獨立的程式，再測試程式與程式之間的連接關係。

5.4 處理錯誤的程式段

處理錯誤的程式段的功用是在錯誤成爲程式的一部份之前將它處理掉，使得程式的執行不致受到影響。它們可以分爲以下幾大項：

1. 輸入
2. 處理
3. 輸出
4. 磁碟

上面的每一項都將在以下幾節中討論，每一個項目都可以放在分開的程式段中處理，同時也都是與程式的其他部份不相關的。

5.4.1 輸入的錯誤

這種型式的錯誤包括了資料型式的錯誤、資料範圍的錯誤、以及對於問題的不正確反應等等。圖5－1就是一個例子，這個程式段是由圖4－7所得來的。

當這個程式要求您輸入一個選擇時（行號950），這個程式將找出不是整數的輸入（行號955），零的輸入（行號970），同時確定輸入值的範圍

（行號 980）；使用這種方法，您就可以十分的確定您所做的選擇是正確的，至於修正錯誤的選擇則完全依賴操作員來做了。

```
      :
55 INPUT "WHICH ONE? ==> ";N :RETURN
60 VTAB 20: HTAB 13: INVERSE :PRINT "ILLEGAL
ENTRY!";CHR$(7): NORMAL
64 FOR D = 1 TO 1000: NEXT D: RETURN
      :

950 VTAB 10:CALL -958:GOSUB 55
955 IF N =INT(N) THEN 970
960 GOSUB 60: GOTO 950
970 IF N = 0 THEN HOME: VTAB 10:PRINT
     "GOODBYE FOR NOW...": END
980 IF N >0 AND N< 4 THEN 990
985 GOSUB 60: GOTO 950
990 ON N GOSUB 2000,4000,6000
     :

1250 VTAB 10: CALL  958
1255 INPUT "ANY CHANGES?(Y/N) ==> ";A$
1260 IF LEFT$(A$,1) = "Y" THEN 1300
1265 IF LEFT$(A$,1) = "N" THEN 1400
1270 GOSUB 60: GOTO 1250
1299 REM CONTINUE PROCESSING
1300 :
      :

1399 REM CONTINUE PROCESSING
1400 :
      :
```

圖5-1 限定輸入範圍的例子

在程式稍後的地方，使用者將被詢問是否有任何需要改正的地方（行號 1255）；在這種情況下，任何被允許的反應都是必須以“Y”或“N”帶頭的，其他的反應（除了CRTL-C，RESET，等等）都將被找出；因此，像是“Y”，“YES”，“YEA”，以及“N”，“NO”，“NADA”等等的回答都是正確的，因爲程式中只查看第一個字（行號1260－1265）。

如果您想得到更好一些控制輸入的方法，那麼您可以使用GET敍述代替

INPUT 敍述；行號 55 就變成了：

```
55 PRINT "WHICH ONE? ==> ";:GET N$:
   PRINT: N = VAL(N$): RETURN
```

而行號 1255 就變成：

```
1255 PRINT "ANY CHANGES(Y/N) ==> ";:
     GET A$: PRINT
```

也許您在看了這個程式之後，您覺得爲了要找出這些錯誤所花費的功夫太大了；但是您必須瞭解一點，那就是您的程式是要在顧客、朋友或是某些您要展示的人之前做示範的。

5.4.2　處理時的錯誤

處理時的錯誤（ Processing errors ）與輸入的錯誤是兩個非常不相像的型式；這種型式的錯誤一般都是發生在程式的邏輯有問題或是因爲一個數字、變數的值超出了範圍或是不合法的時候。

一個值超出了範圍通常發生的情況是在處理一個記憶體位置，而這個記憶體位置並沒有被正確地設定行列值的時候，或是您去處理一個不該去處理的向量或矩陣的位置時發生的。另外一種超出範圍的情況是在圖形的情況下，當您想要處理一個超出螢幕界限之外的點時，就會發生這種情形；這些錯誤都會使您得到錯誤訊息。然而，在高解析度圖形的情況下（high resolution mode)，這個點如果是造型表（ shape table ）的一部份，那麼您只會看到螢幕上一團混亂而不會有任何的錯誤訊息。

一個不合法的數字產生的可能情況有兩種；第一個就是您想用零做除數，APPLESOFT][及整數BASIC 都不能接受這個情況；第二種情形就是您的數字超出了所能允許的範圍，不論是正的或是負的值。

處理過程中經常發生的錯誤就是邏輯的錯誤；這種情形的錯誤，您可能在測試資料時不會發現，但是使用隨機取得的資料去測試就可能找出這個錯誤來了。

不適當的邏輯，錯誤的或不完全的邏輯都會使錯誤產生，並且無法輕易地找出錯誤的所在，通常都必須印出變數值或是使用延遲的方法延緩程式的執行以便查看產生的結果等方法來檢查錯誤，有時候藏在某一個部份的錯誤會使得在前面或是其他部份的執行發生錯誤的情況。

5.4.3 輸出的錯誤

輸出錯誤的原因是不正確的安排螢幕及變數值的顯示，或者是由於您輸入值的不正確或是處理過程中發生錯誤而導致的，這些情形都會使輸出的錯誤極爲明顯。

5.4.4 磁碟的錯誤

我們將在第八章中討論磁碟的錯誤。

5.5 程式設計師的測試工作

在我們前面所討論過的所有項目的測試工作中，我們必須瞭解一件事實，那就是由於程式或程式系統是由程式設計師所設計出來的，因此他（或她）必須擔負起設計測試步驟的責任。因爲只有設計者本人對於程式本身具有較深的瞭解，因此這個工作必須由設計者本人擔負。

我們在這裏所要陳述的一些問題是希望幫助設計者本人在考慮測試步驟時能夠產生更完善的結果。

有一點是非常重要的，那就是程式設計師必須在分析程式或是程式系統的同時發展完成測試的計劃。這樣子才能使測試的步驟隨著程式發展的過程而考慮到每一個層面，當每一個程式段或是每一個系統中的程式被發展出來的同時，測試工作也能夠依照需要而發展完成。

任何的測試步驟都必須是完整而徹底的；進一步的說，它必須深入各層面去探討錯誤發生的可能。

至於最後一項您必須注意的就是測試的步驟應該做爲程式的一部份記錄在程式的記錄內。

5.6 Alpha/Beta測試觀點

在發展您的軟體程式供外界使用的時候，您必須將您的程式置於某一種測試觀點之下，這樣子才能使您的程式適用於某種情況；而這些測試觀點大概可分爲兩種：Alpha 及 Beta 的測試觀點。

Alpha 的測試觀點是著眼於技術方面的問題，它主要是在評估程式技術的品質，軟體的正確性及完整性。下面是它的問題：

這個軟體可靠性高嗎？
它執行的工作是否正確呢？
程式中是否能夠處理錯誤呢？
它是否容易使用？
程式中是否可供使用者選擇所要執行的工作呢？

至於 Beta 的測試則與前面稍有不同，它是著重於使用者的觀點這方面的；它是以使用者的觀點來評估程式，它們所考慮的主要在於程式能夠爲使用者提供多少的服務這個問題。它們的問題如下：

程式做的工作有多好？
它的完整性如何？
是否容易使用？
它執行時可靠嗎？
使用者能夠輕易地改正錯誤嗎？
它是否合乎我的需要？
它是否有用？
它所提供的文字記錄是否清楚又完整？

如果對這些問題的答案都是正確的，那麼您的程式就是一個可靠的程式了。

5.7 總 結

在這一章中我們討論的是測試程式的工作，這個工作使得一個程式能夠正確並且可靠。

我們討論的是程式中每一個部份的測試工作，並且說明在何時及做何種程度的測試；更進一步地，我們將測試程式的方法延伸至程式系統的測試。

程式設計師必須具有的觀點就是他（或她）必須負起測試程式的責任，並且發展出完整及有效的測試步驟。

最後，我們介紹了 Alpha/Beta 的測試觀點，這兩種觀點使得我們以實際的使用立場決定程式所要做的及不要做的工作。

這一章的目的是希望您能瞭解我們花費許多的時間、努力、金錢做測試的工作，它的目的就是要使得我們的程式更好、可靠性更高。

第 6 章

錯誤的處理

6.0 大 綱

這一章將告訴您處理錯誤的重要，因爲錯誤的產生往往是不可預知的。在這一章的討論中，必要的條件是您在輸入程式之前必須將機器清除乾淨，因爲您不知道前面的使用者造成了什麼樣的情況，以及到底是誰用過了這個機器。

爲了要避免您的程式在供其他人使用時產生了損壞的情況，您就必須在您的程式中加入處理輸入錯誤的功能。這個名詞" GIGO "，Garbage-In-Garbage-Out，表示不論輸入或輸出都是沒有用的資料；爲了防止 GIGO 的情形，您只有在輸入的部份確定資料型式及範圍是否正確。

需要記得的是我們所討論的錯誤都是在操作員做輸入動作時產生的；同樣的，有些錯誤是由於程式設計師在設計程式的時候不小心加入的。

6.1 一個乾淨的機器

一個乾淨機器的意思就是說在程式被執行之前您能夠知道所有的開始狀況；當然能夠在任何時刻都能瞭解機器目前的狀況是一個更好的主意，

APPLE 就具有一些簡單而容易使用的指令幫助您達到這個目的。

我們在這裏重複列出圖 2 － 2 中行號 10 的敍述，讓您能夠瞭解一些事實；我們將把這個敍述分爲幾個部份分別說明它們的工作：

```
10 TEXT:CLEAR:POKE 216,0:A = FRE(0)
```

TEXT：爲您處理一些工作；它們是：

* 一將游標放置在文字幕（text screen）的最後一行。
* 一使機器由圖形狀況（graphics mode）返回到文字資料狀況（text mode）。
* 一將螢幕往上捲的範圍設定爲全螢幕的狀況。

CLEAR：執行下列三項重要的工作；它們是：

* 一將所有數值變數（numeric variables）的值設定爲零。
* 一將所有字串變數（string variables）設爲虛值（null）。
* 一將微處理機先入後出暫存器（stack）內所有的返回位置、數字等等資料清除乾淨。

在整數 BASIC 中具有相同功能的是CLR命令。

POKE 216,0：將DOS的錯誤旗標（flag）設爲零。我們在第八章中將討論DOS。

A = FRE(0)：將 APPLESOFT中不再被使用的字串所佔用的位置清除乾淨。這個指令在整數 BASIC 中不能夠使用。

執行過這個敍述之後，您的機器就具有下面這些特點了。

* 一處於文字資料的狀況下。
* 一電腦內的先入後出暫存器已經被清除乾淨供以後呼叫副程式時

* 一所有的變數值都是零，而且字串內都是虛值。
* 一 DOS 所產生的錯誤已經被清除了。
* 一所有空白的記憶體都可以供您使用。
* 一文字幕被設定爲標準的狀況（ 40 行）

您現在已經可以進行您所要做的處理工作了。

6.2　範圍的限定

在許多種情況之下，程式設計師都會要求操作員輸入一個選擇或是在一個範圍內輸入一個數字；在這個時候，程式設計師必須提供一個方法能夠分辨出合法的或是不合法的輸入，這樣的工作也能夠防止不必要的資料被輸入程式中並且能夠使程式的執行更加的可靠。

對於範圍限定的例子我們在 4.7 節中程式架構的部份以及 5.4.1 節中都已經陳述給您了；在 4.7 節中我們將超出 0 到 3 範圍之外的數字或是文字都找出來，這個工作是由行號 935 及 940 的程式行所做的；因此，其他的數字，像是 − 1 或 5 ，都將被找出，而操作員就會看到“ ILLEGAL ENTRY ”的訊息；然後，操作員又可以再度地輸入資料。在 5.4.1 節中我們使用的是同樣的方式，只是加上使用 GET 命令的方法就是了。

另外一個範圍的問題就是當您希望操作員輸入的值是正值或是負值的時候，您必須找出數字的正負號；這也就是說，您必須測試輸入的資料是否具有正確的正負號這個問題。

另外一個問題就是當您希望操作員輸入一個日期的時候；在這種情況下您必須檢查在某一年內所能產生的日期，同時也必須考慮到閏年的問題。圖 6 − 1 就是一個解決的方法。

利用找出正確日期的方法，您就可以計算日期之間的日數，建立正確的日曆，以及計算您的利息了。

```
    :
20 DIM C(12)
30 FOR I = 1 TO 12: READ C(I): NEXT
32 DATA 31,28,31,30,31,30,31,31,30,
    31,30,31
50 GOTO 1000
    :

60 VTAB 20: HTAB 10: INVERSE: PRINT "I
    LLEGAL ENTRY";CHR$(7): NORMAL
65 FOR D = 1 TO 1000: NEXT D: RETURN
    :

1000 HOME: PRINT: PRINT TAB(5) "MORTGA
      GE AMORTIZATION TABLE": PRINT
1010 INPUT "ENTER TODAY'S DATE (MM/DD/
      YY) ==> ";T$
1020 L = LEN(T$):IF L = 8 THEN 1040
1030 C(2) = 28: GOSUB 60:GOTO 1000
1040 Y = VAL ( MID$ (T$,7,2)): M = VAL
      ( LEFT$(T$,2)): D = VAL ( MID$ (
      T$,4,2))
1050 IF INT (Y/4) = Y/4 THEN C(2) = 29
1060 IF M > 0 AND M < 13 THEN 1080
1070 GOTO 1030
1080 IF D > 0 AND D < = C(M) THEN 1100
1090 GOTO 1030
1100 REM continue main here
      :
```

圖6-1 限定資料範圍的例子

這個程式段檢查一個日期的輸入是否正確；程式中做的是下面的工作；

行號 20 — 32 ：設定日曆向量 C 爲 13 個元素，並且將每個月的日數讀入 C(1) 到C(12)內。

行號 50 ：將程式轉移至主程式執行。

行號 60 — 65 ：處理錯誤的程式段，只是將錯誤的訊息顯示在螢光幕上固定的位置，然後做延遲的工作。

行號 1000 ：清除螢幕並將標題顯示出來。

行號 1010 ：要求以固定的格式輸入日期。

行號 1020：檢查輸入日期的長度，如果是正確的，才能夠繼續程式的執行。

行號 1030：呼叫處理錯誤的程式段。

行號 1040：將輸入的日期分爲幾個部份並且轉變成數值。

行號 1050：檢查是否爲閏年，如果是閏年則將二月的天數改爲 29。

行號 1060：檢查輸入的月份是否爲合法。

行號 1080：檢查輸入的日數是否爲合法。

行號 1100：繼續程式的執行。

您現在已經有了一個實際可以被使用的程式段了，您可以將它擴增或修改然後加在您的程式中。

6.3 ONERR敍述

我們可以使用一種非常方便的方法控制一個程式中所產生的不同的錯誤情形，那就是使用 ONERR 敍述。這個指令允許程式設計師分辨及控制由鍵盤輸入或處理時發生的錯誤及磁碟的錯誤，並且可以在不同的副程式中處理這些錯誤。

```
     :
1100 ONERR GOTO 20000:REM TURN ON ONERR
1110 REM INPUT ROUTINES
     :
1895 POKE 216,0: REM TURN OFF ONERR
     :
2000 ONERR GOTO 21000:REM TURN ON ONERR
2010 REM PROCESS ROUTINES
     :
2895 POKE 216,0: REM TURN OFF ONERR
     :
3000 ONERR GOTO 22000:REM TURN ON ONERR
3010 REM DISK PROCESSING
     :
3395 POKE 216,0: REM TURN OFF ONERR
9000 END
```

```
19999 REM INPUT ERROR HANDLER
20000 E = PEEK(222)
      :
      :
20500 RESUME
      :
20999 REM PROCESSING ERROR HANDLER
21000 E = PEEK(222)
      :
      :
21500 RESUME
      :
21999 REM DISK ERROR HANDLER
22000 E = PEEK(222)
      :
      :
22500 RESUME
```

圖6-2 處理錯誤的程式段

不僅僅是錯誤，同時錯誤的型式也可以決定；您可以分別處理程式執行的錯誤，磁碟的錯誤或是輸入時的錯誤；在圖6－2中的程式段就是做這個工作。在附錄E中列有詳細的錯誤碼及錯誤訊息供您參考。

由行號20000到22500是處理不同錯誤的許多程式段；您可以使用存有錯誤碼的變數E，或是依照您的選擇，轉移到您所希望轉移到的處理錯誤的程式段去。您也許希望做下面的事：

* 一清除一個變數的值，
* 一要求輸入另外一個值，
* 一要求一個正確的反應，
* 一修正一個磁碟命令，
* 一建一個新的檔，
* 一要求輸入失去的資料，
* 一重新使系統存入記憶體內，
* 一警告操作員，或是
* 一停止程式的執行。

顯然還有其他許多種的工作可以執行，這完全依照您程式中的需求而決定了，在這裏我們只是告訴您一些可能做的工作而已。現在我們希望您參考在附錄A中所描述的 ONERR 及 RESUME命令。

6.4　輸入的限定

處理錯誤程式段的目的就是要在錯誤成爲程式的一部份之前將它找到並處理完畢，以免妨礙了程式的執行；這種處理錯誤的程式段包含了許多種方式，它們分別在第五章及這一章中陳述給您。

我們現在再度討論限定輸入範圍的原因就是因爲它是非常重要的；畢竟，如果您輸入的是錯誤的資料，那麼您的程式必定失敗；因此，請您務必要能夠瞭解防止輸入錯誤的方法以及如何處理輸入的錯誤。

6.5　總　結

在這一章中，我們討論的是處理錯誤的方法以及程式中可能發生錯誤的地方。

我們也討論了如何建立一個乾淨機器的方法，因此您在以後設計您的程式時就可以使用了。這個方法包含了下面的指令：

```
TEXT, CLEAR, POKE 216,0, A = FRE(0)
```

其後一節中討論的是範圍限定及處理的方法，這就是第五章討論結果的實際應用了。

6.3節討論的則是 APPLESOFT中 ONERR 及 RESUME的指令；我們利用例子告訴您如何使用不同的程式段處理不同的錯誤情形。

至於最後一節則重新的討論了輸入限定的問題，那是想要提醒您輸入工作實在是程式中非常重要的一環。

第7章 應用程式

7.0 大 綱

這一章討論使用Apple][解決一些應用程式的問題；我們將告訴您不同的排列(sorts)及搜尋(searches)的方法；像這些型式的程式都是非常重要的，因爲電腦經常被用來由檔中找出資料，或是由檔中改變資料；由於目前尋找資料時找出的資料必定與上一次的所要求的對象不相同；故而，我們必須重新搜尋我們所要的資料並且重新做排列的工作。

我們將討論一些其他的應用程式，它們都是非常有趣而且可以做爲大型程式的副程式來使用的。我們希望我們介紹這些程式對您設計程式而言能有很大的助益。

7.1 排列的程式

排列資料是一個非常費時的工作；在這一節中，我們將告訴您許多種不同的排列方法，並且討論它們各別的特性。有一種排列的方法可以分別應用在不同的情況下，完全依照資料的數量及資料的型式等等因素而決定。排列資料的時候可以使用數值的或是字串的資料型式，只是字串型式的排列比較費時就是了。

7.1.1 Bubble的排列方式

第一個排列的例子就是交換方式的排列方法，作業的時候需要將資料的一個位置轉放到另一個位置；這也就是說，我們對於不合次序的一對資料做交換位置的工作。

最簡單的方法就是比較連續的兩個資料，如果它們不依次序排列，那麼就交換這兩個資料的位置，使它們按照次序排列；然後依次往下做比較及交換的工作，等到所有的比較及交換的工作都完成了之後，我們就可以得到按照次序排列的資料列了。圖 7 — 3 就是資料由左往右移動的情形，最後得到的就是排列完成後的結果。

在圖 7 — 2 中的程式將資料往後面移動，這是因爲我們排列的方式是將大的資料移往後面的位置，最後的結果就是依照它們的數值大小順序的排列。

下面就是這個排列的過程：

假設：R1<=R2<=.....Rn是要排列順序的記錄（records）：在排列完成之後，它們將依 R1,R2,.....Rn 的順序排列。

1. U=n ；U 是所有資料的上限，我們使用它代表目前我們尚不知道的記錄數目。
2. 設定 SW=0 ：執行第 3 步驟（ j = 1,2,.....U-1 ），再往第 4 步。
3. 比較 Rj:Rj+1 ：如果 Rj>Rj+1 則做交換的工作，並把 SW 設定爲 1 。
4. 如果 SW=0 則結束排列的過程，否則 U=U-1 並往第 2 步繼續執行。

圖7-1 Bubble排列方式的程序

```
1   REM *** BUBBLE SORT ***
2   REM *
3   REM * WRITTEN BY:JL CAMPBELL
4   REM *       DATED:08/01/80
5   REM *
6   REM * COMPUTER:APPLE II
7   REM * LANGUAGE:APPLESOFT II
8   REM *
9   REM *******************
10  TEXT : CLEAR : POKE 216,0:A =  FRE (0)
20  HOME : PRINT : PRINT  TAB( 10)"*** BUBBLE SORT
***": PRINT
30  READ N
32  DATA  10
40  DIM X(N)
50  FOR I = 1 TO 10: READ X(I): NEXT I
60  DATA   99,56,73,1,4,18,67,9,3,0
99  REM * ENTER SORT
100 N1 = N - 1:C = 1
110 SW = 0
120  FOR I = 1 TO N1
130  IF X(I) <  = X(I + 1) THEN 220
135  REM * EXCHANGE
140 T = X(I)
150 X(I) = X(I + 1)
160 X(I + 1) = T
170 SW = 1
175  PRINT C;"- ";
180  FOR J = 1 TO N: HTAB 3 * (J - 1) + 4
190  PRINT X(J);
200  NEXT J: PRINT
210 C = C + 1
220  NEXT I
230  IF SW = 0 THEN 250
240 N1 = N1 - 1: GOTO 110
250  PRINT
260  FOR I = 1 TO N: HTAB 3 * (I - 1) + 4
270  PRINT X(I);
280  NEXT I: PRINT
300  PRINT : PRINT "THAT'S ALL FOLKS!"
310  END
```

圖7-2 Bubble方式的排列程式

有一點必須瞭解的，那就是 Bubble 的排列方式執行的速度非常慢，當資料的數目增加時，排列的時間就會加長許多，這是因爲比較及交換的次數也同樣增加的原因。

```
               *** BUBBLE SORT ***

1-   56 99 73 1  4  18 67 9  3  0
2-   56 73 99 1  4  18 67 9  3  0
3-   56 73 1  99 4  18 67 9  3  0
4-   56 73 1  4  99 18 67 9  3  0
5-   56 73 1  4  18 99 67 9  3  0
6-   56 73 1  4  18 67 99 9  3  0
7-   56 73 1  4  18 67 9  99 3  0
8-   56 73 1  4  18 67 9  3  99 0
9-   56 73 1  4  18 67 9  3  0  99
10-  56 1  73 4  18 67 9  3  0  99
11-  56 1  4  73 18 67 9  3  0  99
12-  56 1  4  18 73 67 9  3  0  99
13-  56 1  4  18 67 73 9  3  0  99
14-  56 1  4  18 67 9  73 3  0  99
15-  56 1  4  18 67 9  3  73 0  99
16-  56 1  4  18 67 9  3  0  73 99
17-  1  56 4  18 67 9  3  0  73 99
18-  1  4  56 18 67 9  3  0  73 99
19-  1  4  18 56 67 9  3  0  73 99
20-  1  4  18 56 9  67 3  0  73 99
21-  1  4  18 56 9  3  67 0  73 99
22-  1  4  18 56 9  3  0  67 73 99
23-  1  4  18 9  56 3  0  67 73 99
24-  1  4  18 9  3  56 0  67 73 99
25-  1  4  18 9  3  0  56 67 73 99
26-  1  4  9  18 3  0  56 67 73 99
27-  1  4  9  3  18 0  56 67 73 99
28-  1  4  9  3  0  18 56 67 73 99
29-  1  4  3  9  0  18 56 67 73 99
30-  1  4  3  0  9  18 56 67 73 99
31-  1  3  4  0  9  18 56 67 73 99
32-  1  3  0  4  9  18 56 67 73 99
33-  1  0  3  4  9  18 56 67 73 99
34-  0  1  3  4  9  18 56 67 73 99

     0  1  3  4  9  18 56 67 73 99

THAT'S ALL FOLKS!
```

圖7-3　Bubble排列方式的輸出

請注意數字 18 及 67 由於作業上的需要都必須先往左邊移動，然後才向右移到它最後的位置。

7.1.2 插入的排列方式

一個有趣並且非常重要的排列方法就是插入法(insertion sort)；它將要加入的資料直接地插入先前已經排列的資料列內，使用的方法是將資料或指標移開讓插入的資料排列，當插入的位置被找到之後，所有在其後的資料都往右（或下）移動。讓出空間供插入的資料放置。

下面就是這種排列方式的程序：

假設： R1,R2.....Rn 是要被重新排列的記錄（records），在排列完成之後，它們的key將依照 K1<= K2 <=.... <= Kn 順序排列。

1. j 的廻圈，在 j = 2,3,...,n 時執行 2 到 5 步，然後結束這個程序。
2. 設定 i = j-1: K =Kj: R =Rj
3. 比較 K與 Ki ：如果 K >= Ki 則往第 5 步轉移。
4. 移動 Ri ，並將 i 減 1 ；設定 Ri+1 = Ri : i =i-1 ：如果 i > 0 則往第 3 步。
5. 設定 Ri+1 = R

圖7-4 插入法的程序

```
1 REM *** INSERTION SORT ***
2 REM *
3 REM * WRITTEN BY:JL CAMPBELL
4 REM *        DATED:06/19/81
5 REM *
6 REM * COMPUTER:APPLE II
7 REM * LANGUAGE:APPLESOFT II
8 REM *
9 REM ***********************
```

```
10  TEXT : CLEAR : POKE 216,0:A =  FRE (0)
20  DIM X(50),Y(50)
30  READ NR,NI
32  DATA   5,8
40  FOR I = 1 TO NR: READ X(I): NEXT
42  DATA  7,9,12,16,20
44  FOR I = 1 TO NI: READ Y(I): NEXT
46  DATA   8,10,25,5,3,30,18,40
50  HOME : PRINT : PRINT  TAB( 9)"*** INSERTION SORT
***": PRINT
100  FOR I = 1 TO NR: HTAB 3 * (I - 1) + 1
110  PRINT X(I);
120  NEXT I: HTAB 22: PRINT "<== ORIGINAL LIST"
130  GOSUB 2000
140  PRINT : PRINT "THAT'S ALL FOLK!"
150  END
2000  REM  * INSERTION SORT
2010 UB = NR
2020  FOR J = 1 TO NI
2030 UB = UB + 1
2040  IF Y(J) >  = X(UB - 1) THEN IP = UB: GOTO 2170
2050  FOR I = UB - 1 TO 1 STEP  - 1
2060 IP = I
2100  IF Y(J) >  = X(IP) THEN 2210
2110  NEXT I
2140  FOR K = UB TO 2 STEP  - 1
2150 X(K) = X(K - 1): NEXT K
2160 IP = 1
2170 X(IP) = Y(J): GOTO 2260
2210  FOR K = UB TO IP + 1 STEP  - 1
2220 X(K) = X(K - 1)
2230  NEXT K
2240 X(IP + 1) = Y(J)
2260  FOR L = 1 TO UB: HTAB 3 * (L - 1) + 1
2270  PRINT X(L);
2280  NEXT L: PRINT
2290  NEXT J
2310  RETURN
```

圖7-5 插入法的程式

```
        *** INSERTION SORT ***

7  9  12 16 20                                    <= 原先資料列
7  8  9  12 16 20                                 <= 插入   8
7  8  9  10 12 16 20                              <= 插入   10
7  8  9  10 12 16 20 25                           <= 插入   25
5  7  8  9  10 12 16 20 25                        <= 插入   5
3  5  7  8  9  10 12 16 20 25                     <= 插入   3
3  5  7  8  9  10 12 16 20 25 30                  <= 插入   30
3  5  7  8  9  10 12 16 18 20 25 30               <= 插入   18
3  5  7  8  9  10 12 16 18 20 25 30 40            <= 插入   40

THAT'S ALL FOLKS!
```

我們現在必須指出這種插入法對於短的資料列(50個記錄)是非常有效的;然而,當資料的數目增加時,插入的時間會增加,這是因爲我們在插入之前都必須做一個線性搜尋(linear search)的原因。

同時只要插入的位置 IP ,被找到了,那麼,所有在這個位置右邊的資料都必須往右邊移動一個位置——由這個資料列的最後一個資料開始,到與 IP 相接的資料爲止;然後,才將新的記錄或值插入。

在這個程序中至少有一個地方可以使程式的執行速度加快,那就是在搜尋的部份;如果我們用二分法的搜尋 binary search ——請參看7.2.2節,代替線性搜尋一行號 2050 到 2110 ,那麼速度就會加快許多。

另外一個增加這個程式速度的方法就是根本不要移動記錄;在這種情況下,您就必須建立代表每個記錄的key及指標,那麼您在排列的時候移動的只是記錄的key及指標而不是記錄了。請您參看關於資料基(database)或是資訊系統方面的書。

7.2　搜尋資料

在許多程式中搜尋(Searching)是一件最費時的工作，如果能夠將一個有效且快速的搜尋程式段加到應用程式內，那將會大幅度的增加應用的程式的市場銷售率。

搜尋可以分爲許多種方法，它們可能是：

* *—內部的或外部的
* *—靜態的或動態的
* *— Key的比較或是依照 key的特性搜尋

內部的搜尋是在電腦的記憶體內完成的，這種情形的執行速度要比在磁碟上做搜尋快得多。

靜態的搜尋表示所要找的資料列是穩定的（不變的），這種情形使得搜尋的工作即快又有效。至於動態的搜尋則表示您想要改變資料列中的資料，不論是插入，消除或更改都屬於這種情形。

第三種情形則是您搜尋時是依照key的值做爲比較，或是依照key的數值特性，數元型式做爲比較。

7.2.1　線性搜尋

由資料的起始開始並一直找到相合的地方爲止，這就是線性搜尋的過程；線性搜尋可以說是一個最簡單也最容易瞭解的搜尋方式。

下面就是線性搜尋的程序：

假設有一列的記錄， R1,R2,.....Rn ，它們的key分別是K1,K2.....Kn ；這個程序搜尋K這個元素，我們假設 n>=1

1. 設定 I = 1
2. 如果 K = Ki 則這個程序成功的結束。
3. I = I + 1
4. 如果 I <= n ，則往第二步執行，否則；
5. 程序不成功的結束。

圖7-6 線性搜尋的程序

請注意這個程序可以成功或不成功的結束，這對大多數的搜尋程序來講都是眞實的情況。

接下來在圖 7 — 7 中的程式使您能夠依照記錄的值做搜尋或是依照代表記錄的數字直接處理記錄值。

線性的搜尋是非常直接的，您由第一個記錄開始做比對的工作，直到您找到了符合的記錄爲止；這是由行號 2000 區域的程式段執行的。

如果您的資料列不長— 50 個記錄左右，那麼搜尋的時間是可以忍耐的；平均來說，您必須搜尋資料列中 1/2 的資料才能找到您所需要的記錄；因此，如果將您的資料列加大，那麼它所需要的時間也就隨之而增加。一個擁有 10,000 個記錄的資料列，您每次要找資料時平均必須處理 5,000個記錄。

```
1   REM *** LINEAR SEARCH ***
2   REM *
3   REM * WRITTEN BY:JL CAMPBELL
4   REM *        DATED:08/01/80
5   REM *
6   REM * COMPUTER:APPLE II
7   REM * LANGUAGE:APPLESOFT II
8   REM *
9   REM *******************
10   TEXT : CLEAR : POKE 216,0:A =  FRE (0)
20   DIM Z(10)
30   FOR I = 1 TO 10: READ Z(I): NEXT I
40   DATA   5,7,2,9,1,4,3,8,6,0
50   HOME : PRINT : PRINT  TAB( 9)"*** LINEAR SEARCH
***": PRINT
```

```
60   PRINT "MENU:": PRINT "----"
70   PRINT  TAB( 5)"1- BY RECORD #": PRINT
80   PRINT  TAB( 5)"2- BY VALUE": PRINT
90   VTAB (10): CALL  - 958: INPUT "WHICH ONE? ==> ";N
100   IF N > 0 AND N < 3 THEN 120
110   VTAB (20): HTAB 10: INVERSE : PRINT "ILLEGAL
ENTRY": NORMAL : PRINT  CHR$ (7)
115   FOR L = 1 TO 1000: NEXT L: GOTO 90
120   ON N GOSUB 1000,2000
130   VTAB (20): INPUT "ANOTHER CALCULATION?(Y/N) ";A$
140   IF  LEFT$ (A$,1) = "Y" THEN 50
150   VTAB (4): CALL  - 958: VTAB (10)
160   PRINT "GOODBYE FOR NOW...": END
1000   VTAB (4): CALL  - 958: INPUT "WHICH RECORD? ==>
";R
1010   VTAB (6): PRINT "RECORD #";R;" = ";Z(R)
1020   RETURN
2000   VTAB (4): CALL  - 958: INPUT "WHICH VALUE? = ";V
2010   FOR I = 1 TO 10
2020   IF Z(I) = V THEN 2100
2030   NEXT I
2040   VTAB (6): PRINT "*** NO SUCH VALUE! ***"
2050   RETURN
2100   VTAB (6): PRINT "RECORD #";I;" = ";Z(I)
2110   RETURN
```

圖7-7　線性搜尋的程式

7.2.2　二分法的搜尋

二分法搜尋(binary　search)是以2的次方爲基楚的，在這種情況下，我們使用2做爲因子，使得我們在每次循環的時候能夠將搜尋的次數減半。第一次執行的時候就會將資料列的一半消除了(如果沒有相等的話)，在第三次執行的時候，您只需要尋找剩下的1/8的記錄就可以了。

下面就是它的程序：

假設有一列記錄， R1,R2,.....Rn ，它們的key是依照漸增的次序排列的， K1,K2,.....Kn 。這個程序搜尋一個設定的元素，K 。

1. 設定 l = 1, u = n
2. 如果 u < l 則不成功的結束。
3. 設定 i = int((l + u)/2)，找出中點。
4. 如果 K > Ki 則前往第 8 步執行。
5. 如果 K < Ki 則前往第 7 步執行。
6. 成功地結束。
7. u = i - 1: goto 2
8. l = i + 1: goto 2

圖7-8　二分法搜尋的程序

下面的程式使您能夠依照記錄的值或key找到這個記錄。

這個程序有兩種可能的情況存在，它們顯然就是成功或失敗的情況；失敗的情形就是當上限值小於下限值時發生的，這種情況發生時就表示雖然您已經搜尋過整個的資料列了，但是並沒有發現您所要找的key值。

二分法的搜尋對於尋找資料列中的某一個資料非常有效，尤其在這個資料列很大的時候；一個擁有1000個記錄的資料列您只需要查看其中的 10 個記錄就可以找到您要的資料了。

```
         !
     *********
     * u = n *
     * l = l *
     *********
         !
       *****
      *     *    是
+--->* u < l *------------>   失敗
!     *     *
!      *****
!        ! 不是
!    *********
!    *       *
!    * 設定 i *
!    *       *
!    *********
!        !
!      *****
!     *     *    是
!    * K >Ki *------------>+
!     *     *              !
!      *****               !
!        ! 不是              !
!      *****               !
!     *     *    是         !
!    * K <Ki *---------->+ !
!     *     *            ! !
!      *****             ! !
!        ! 不是            ! !
!        !               ! !
!       成功               ! !
!                        ! !
!    *********           ! !
!    *       *           ! !
+<---* u=i-l *<----------+ !
!    *       *             !
!    *********             !
!                          !
!    *********             !
!    *       *             !
+<---* l=i+l *<------------+
     *       *
     *********
```

圖7-9 二分法搜尋的流程圖

```
1 REM ***** BINARY SEARCH *****
2 REM *
3 REM * WRITTEN BY:JL CAMPBELL
4 REM *       DATED:06/07/81
5 REM *
6 REM * COMPUTER:APPLE II
7 REM * LANGUAGE:APPLESOFT II
8 REM *
9 REM **************************
10  TEXT : CLEAR : POKE 216,0:A =  FRE (0)
20  DIM X(100)
30  READ NR
32  DATA  30
40  FOR I = 1 TO 20: READ X(I): NEXT
42  DATA  1,3,5,7,8,9,10,12,16,20
44  DATA  24,30,33,35,38,39,40,50,52,60
50  HOME : PRINT : PRINT  TAB( 9)"*** BINARY SEARCH
     ***": PRINT
60  PRINT "MENU:": PRINT "----"
70  PRINT  TAB( 5)"1- BY RECORD #": PRINT
80  PRINT  TAB( 5)"2- BY VALUE": PRINT
90  VTAB 10: CALL  - 958: INPUT "WHICH ONE? ==> ";N
100  IF N > 0 AND N < 3 THEN 120
110  VTAB 20: HTAB 10: PRINT "ILLEGAL ENTRY!": GOTO 90
120  ON N GOSUB 1000,2000
130  VTAB 20: INPUT "ANOTHER SEARCH?(Y/N) ";A$
140  IF  LEFT$ (A$,1) = "Y" THEN 50
150  VTAB 4: CALL  - 958: VTAB 10
160  PRINT "GOODBYE FOR NOW....": END
1000  VTAB 4: CALL  - 958: INPUT "WHICH RECORD? ==> ";R
1010  VTAB 6: PRINT "RECORD #";R;" = ";X(R)
1020  RETURN
2000  VTAB 4: CALL  - 958: INPUT "WHICH VALUE?  ==> ";V
2010 L = 1:U = NR
2020  IF U < L THEN 2200
2030 I =  INT ((U + L) / 2)
2040  IF V < X(I) THEN 2070
2050  IF V > X(I) THEN 2080
2060  IF V = X(I) THEN 2100
2070 U = I - 1: GOTO 2020
2080 L = I + 1: GOTO 2020
2100  VTAB 6: CALL  - 958
2110  PRINT "RECORD #";I;" = ";X(I)
2120  RETURN
2200  VTAB 6: CALL  - 958
2210  PRINT "*** NO SUCH VALUE ***"
2230  RETURN
```

圖7-10　二分法搜尋的程式

```
          1         2
12345678901234567890  20個記錄
!                   !
L---------+---------U
          ^         !  I = 10:  第1次探查
          L----+----U
               ^    !  I = 15:  第2次探查
               L--+-U
                  ^ !  I = 18:  第3次探查
                  L+U
                   ^   I = 19:  第4次探查
                   !
                     成功
```

要想執行二分法的搜尋，您必須使您的資料具有下列的特性；

* 一依照漸增的順序排列。
* 一記錄的數目必須知道。

現在讓我們假定您找到key值爲 52 的記錄，由圖 7－10 的 data 敍述中您可以發現這個記錄位於 X 向量的第 19 個位置；對於一個擁有 20 個記錄的資料列而言，最壞的情況就是經過 5 次的探查就可以找到我們所要的資料了。這是由 2 ^ 4 = 16 ，2 ^ 5 = 32 這樣的計算而得來的事實。

由上面的圖中我們知道，在第一次的探查完成之後，1/2的記錄就被消除了；在第二次探查之後，資料列中的記錄，就有 3/4 被消除了，每一個探查都使得原先資料列中 1/2的資料被消除；在這種情況下您就可以很快地搜尋到您所要的資料。請注意下限(L)移動的情形。

上面的圖中我們發現在 4 次探查之後，我們就找到了所需要的資料，而在線性搜尋中則需要 19 次的探查才能找到這個資料。

7.2.3　字串的搜尋

這個搜尋實際上就是線性的搜尋；因此，它的程序與 7.2.1 的相同。

```
1  REM *** STRING SEARCH ***
2  REM *
3  REM * WRITTEN BY:JL CAMPBELL
4  REM *       DATED:08/01/80
5  REM *
6  REM * COMPUTER:APPLE II
7  REM * LANGUAGE:APPLESOFT II
8  REM *
9  REM *******************
10  TEXT : CLEAR : POKE 216,0:A =  FRE (0)
20  HOME : PRINT : PRINT  TAB( 9)"*** STRING SEARCH
***": PRINT
30  DIM X(30)
40 A$ = " ABCDEFGHIJKLMNOPQRSTUVWXYZ.?!"
50  FOR I = 1 TO 30:X(I) = 0: NEXT I
100  INPUT "ENTER STRING = ";S$
110  FOR I = 1 TO  LEN (S$)
120  FOR J = 1 TO 30
130  IF  MID$ (A$,J,1) =  MID$ (S$,I,1) THEN X(J) =
X(J) + 1
140  NEXT J,I
150  VTAB (7)
160  FOR I = 1 TO 30 STEP 3
170 T = 1
180  FOR J = I TO I + 2: HTAB 8 * (T - 1) + 5
190  PRINT  MID$ (A$,J,1);" = ";X(J);
200 T = T + 1
210  NEXT J: PRINT
220  NEXT I: PRINT
230  PRINT "THAT'S ALL FOLKS!"
240  END
```

圖7-11　字串搜尋的程式

這個程式實際上搜尋一個字串，計算出每個字發生的次數並且將結果顯示在螢光幕上，這種字串的搜尋一般被使用在尋找名字或是地址等等方面。

7.3　技術與格式

到目前為止您所見過的程式都具有作者的技術與獨自的格式，我們並不認為這格式是獨一的，我們所想要表示的只是我們認為較合適的方式，我們希望您能夠從這些程式中使您得到設計程式的興趣，並且將這些格式加入您的程式中，我們也希望使用這種方法使您能夠設計出更好的程式來。

每個人設計程式的格式都是不大相同的，但是只要這種格式能與結構與分析形成良好的搭配，那麼您就是成功的設計師了。我們也說明了一些事實，這些事實對您瞭解一個程式及組織一個程式的幫助都相當大，我們現在希望將它們加入這一節的討論中，因為您已經見過許多程式的例子了。

1. 要被讀取的 Data 資料應該跟在 READ 的後面，而不要放在程式的結尾。這麼做將使您在檢查程式時能夠很快的知道那一個資料是要被讀取的。叁看圖 7－1，7－5，7－7及7－10。
2. 所有的字串及常數都應該在程式中早一些宣告，這樣子會使得這些值能夠較早地存入變數表內，而在處理時能夠快速地被找到。請參看圖 7－11。
3. 在處理程式之前必須先將機器清除乾淨，參看圖 2－2，7－1，7－5，7－8，7－10及7－12。
4. 依照下列的次序執行程式。

   a. 輸入所有的資料。
   b. 進行處理的工作。
   c. 輸出結果。

5. 程式設計的方式依照由上往下的結構型式設計。參看圖 7－1，7－5，7－12 等例子。
6. 標明程式的目的。參看圖 7－1，7－5，7－8，7－11及7－12。

7. 將程式做成文件資料，這樣您才能夠在以後再度的瞭解它。
8. 使您的程式盡量地能夠快速執行。
9. 不要自做聰明的做一些擾亂程式邏輯的工作。
10. 使用的變數名稱具有邏輯上的意義。參看圖 4 － 7 ， 7 － 7 ， 7 － 10 ， 7 － 11 等例子。
11. 除了上面的幾種情況外，您必須不害怕嘗試的工作，盡量地去實驗也許您會發現一個非常有用的工具。

7.4 各式各樣的程式

這一節包含了許多的程式及副程式，我們認爲它們都有極大的價值，因此在這裏介紹給您供您使用。

如果您有一些副程式或是程式具有某些特定的功能，那麼您就可以將它們存在磁片內，等到在其他的程式中需要使用這些功能的時候，您就可以利用 Apple][提供的 RENUMBER 程式，將它們加入您的程式中供您使用了。

7.4.1 分解一個數字

這個程式就是將一個數字分開，並將每個被分開的數字放置到適當的位置去。就它本身的價值而言似乎並沒有什麼意義，但是當您想要指定小數位置時，它將是非常有用的。

這個程式與 Horners Rule 是相反的情況，它能夠處理 12 個位置的數字。

您也許布望將它改變符合您的需要，這個程式希望您輸入您所要求的精確度的位數，然後再輸入數字本身，但是並不需要輸入小數點，產生的結果就是依照您要求的精確度而排列的數字。像這樣的程式您可以將它修改爲在數目前加上符號及其他一般帳上所需要的符號，而對操作員來說同時也省

卻了句點及小數點的輸入工作，

```
1   REM *** DECOMPOSE ***
2   REM *
3   REM * WRITTEN BY:JL CAMPBELL
4   REM *       DATED:08/01/80
5   REM *
6   REM * COMPUTER:APPLE II
7   REM * LANGUAGE:APPLESOFT II
8   REM *
9   REM *******************
10   TEXT : CLEAR : POKE 216,0:A =  FRE (0)
20   DIM Z(12)
30   HOME : PRINT : PRINT  TAB( 7)"*** DECOMPOSE A
NUMBER
     ***": PRINT
40   INPUT "HOW MANY DECIMAL PLACES? = ";D: PRINT
50   INPUT "ENTER THE NUMBER          = ";X
60 SW = 0
70 X1 =  INT (X + 0.5)
80   FOR I = 1 TO 12:Z(I) = 0: NEXT I
90   FOR I = 12 TO 1 STEP  - 1
100 X2 =  INT (X1 / 10):X3 =  INT (X2 * 10)
110 Z(I) =  INT (X1 - X3)
120 X1 = X2
130  NEXT I
140  VTAB (8): PRINT "DECOMPOSITION = ";
150  FOR I = 1 TO (12 - D)
160  IF SW = 0 AND Z(I) = 0 THEN 190
170  PRINT Z(I);" ";
180 SW = 1
190  NEXT I
200  PRINT ".";
210  FOR I = (12 - D + 1) TO 12
220  PRINT Z(I);" ";
230  NEXT I
240  PRINT : PRINT
250  PRINT "THAT'S ALL FOLKS!"
260  END
```

圖7-12　分解數字的程式

7.4.2 Horners法則

Horners Rule 就是解多項式 polynomial 方程式的一種方法（變數的係數必須已知）。多項式方程式可以以許多種的方式來解。

具 n 次的多項式方程式一般寫成下面的型式；

```
p(X) = An*X^n +....+ A2*X^2 + A1*X + A0   An <> 0
```

在使用重複計算的方法中，我們必須計算這個方程式許多次，因此對於計算的速度我們必須儘量使它快速。

將每一項都分開計算再將它們加在一起的計算方式，是一個極沒有效率的方法，因爲這種方式的計算共需要執行 n 次的加法，以及 n*(n+1)/2 次的乘法。

然而，如果您觀察上面的方程式，除了A0之外，您可以發現它們都可以提出因數，也就是將 x 移出，那麼產生的結果就寫成n-1次了，而這個結果仍然可以再提出 x，如此繼續下去，則我們可以看出 p(X) 可以以巢狀的格式表示出來。

假如我們有這樣的一個被多項式程式：

```
p(X) = 1*X^4 + 2*X^3 + 3*X^2 + 4*X^1
```

而我們想計算 x = 2 的值。

因此，

```
p(2) = (((1*2 + 2)*2 +3)*2 + 4)*2 = 52
```

這個解法的程序如下面一般；

假設我們已知多項式的 A1，...，An的係數值及 X 值。

1. 設定 P = A1
2. FOR I = 2 TO n　　執行；
3. P = P * X + Ai

這個方法在數值分析方面有很大的功用，它可以將一個二進位的數字轉換成十進位的數字或是以其他數字為基底的數字，您可以試著使用這個方法將二進位數字 11000101 轉換成十進位數字 197 。

```
1   REM *** HORNERS RULE ***
2   REM *
3   REM * WRITTEN BY:JL CAMPBELL
4   REM *      DATED:08/01/80
5   REM *
6   REM * COMPUTER:APPLE II
7   REM * LANGUAGE:APPLESOFT II
8   REM *
9   REM ********************
10   TEXT : CLEAR : POKE 216,0:A =  FRE (0)
20   DIM Z(5)
30   FOR I = 1 TO 5: READ Z(I): NEXT I
40   DATA  1,2,3,4,5
50   HOME : PRINT : PRINT  TAB( 10)"*** HORNERS RULE
      ***": PRINT
60   PRINT "DECOMPOSITION = ";
70   FOR I = 1 TO 5: PRINT Z(I);" ";
80   NEXT I: PRINT : PRINT
90 X = Z(1)
100   FOR I = 2 TO 5:X = X * 10 + Z(I): NEXT I
110   PRINT "FINAL NUMBER = ";X
120   PRINT : PRINT "THAT'S ALL FOLKS!"
130   END
```

圖7-13　Horners方法

7.4.3 求極大及極小值

這個程式在您由鍵盤輸入的一組數字中找出極大及極小值。

```
1   REM *** MAX & MIN ***
2   REM *
3   REM * WRITTEN BY:JL CAMPBELL
4   REM *        DATED:07/21/80
5   REM *
6   REM * COMPUTER:APPLE II
7   REM * LANGUAGE:APPLESOFT II
8   REM *
9   REM *******************
10   TEXT : CLEAR : POKE 216,0:A =  FRE (0)
20   DIM X(100)
50   HOME : PRINT : PRINT  TAB( 9)"*** FIND MAX & MIN
      ***": PRINT
60   INPUT "HOW MANY NUMBERS? = ";N%
70   FOR I = 1 TO N%
80   VTAB (6): CALL  - 958
90   PRINT "ENTER NUMBER #";I;: INPUT " = ";X(I)
100   NEXT I
110   VTAB (4): CALL  - 958: VTAB (10)
120   HTAB 10: INVERSE : PRINT "... PROCESSING ...":
NORMAL
130 MX = X(1):MN = X(1)
140   FOR I = 2 TO N%
150   IF X(I) >  = MX THEN MX = X(I)
160   IF X(I) <  = MN THEN MN = X(I)
170   NEXT I
180   VTAB (4): CALL  - 958
190   VTAB (6): PRINT "MAXIMUM NUMBER = ";MX
200   VTAB (8): PRINT "MINIMUM NUMBER = ";MN
210   VTAB (10): INPUT "ANOTHER PROBLEM?(Y/N) ";A$
220   IF  LEFT$ (A$,1) = "Y" THEN 50
230   VTAB (12): PRINT "GOODBYE FOR NOW...": END
```

圖7-14 求極大，極小值的程式

7.4.4 Soundex方法

在搜尋中一種有趣的方式就是在大的資料群中搜尋姓氏的方法；爲了要使這種搜尋能夠快速及精確地執行並且能夠找出具有某些特性的名字，Margaret K. Odell及 Robert C. Russell 發展出了"Soundex"方法。下面就是這個方法的程序：

1. 取得每個名字的第一個字母。
2. 在其他的位置上去除 a,e,h,i,o,u,w及y的出現機會。
3. 對於在第一個字之後的其他字母給予代表的數字。

```
1 = b,f,p,v
2 = c,g,j,k,q,s,x,z
3 = d,t
4 = l
5 = m,n
6 = r
```

4. 如果在原先的名字中有連續兩個或兩個以上的字具有同樣的數字碼，那麼除了第一個之外其餘的都取消掉。
5. 將名字改換成這種型式：
   "字母、數字、數字、數字"
6. 在尾部加上零（如果少於三位數字）。
7. 將右邊的數字去掉（超過三位數）

圖7-15 SOUNDEX程序

當您使用這種方式來轉換名字時，產生的 soundex 碼就變成到表中取資料的key了。

```
1 REM ***** SOUNDEX *****
2 REM *
3 REM * WRITTEN BY:JL CAMPBELL
4 REM *      DATED:02/15/78
5 REM *
6 REM * LANGUAGE:APPLESOFT II
7 REM * COMPUTER:APPLE II
8 REM *
9 REM *******************
10 NR$ = "01230120022455012623010202"
20  HOME : PRINT : PRINT  TAB( 12)"*** SOUNDEX ***":
PRINT
30  INPUT "ENTER LAST NAME = ";LN$
40  GOSUB 2000
50  PRINT : PRINT "SOUNDEX CODE = ";S$
60  VTAB (20): INPUT "ANOTHER NAME?(Y/N) ";A$
70  IF  LEFT$ (A$,1) = "Y" THEN 20
80  END
2000  REM * SOUNDEX SUBR
2010 L$ = " "
2020 S$ =  LEFT$ (LN$,1)
2030  IF  LEN (LN$) < 2 THEN 2200
2040  FOR I = 2 TO  LEN (LN$)
2050 E$ =  MID$ (LN$,I,1)
2060 E =  ASC (E$) - 64
2070  IF E > 26 OR E < 1 THEN 2160
2080  REM   SELECT CODE
2100 K$ =  MID$ (NR$,E,1)
2120  IF K$ = L$ OR K$ = "0" THEN 2160
2140 S$ = S$ + K$
2150  IF  LEN (S$) > 3 THEN 2200
2160 L$ = K$
2170  NEXT I
2200 S$ =  LEFT$ (S$ + "000",4)
2210  RETURN
```

圖7-16 Soundex程式

7.4.5 Print Using

在這些電腦中您可以使用 PRINT USING 敍述定義您印出數字的格式但是在整數 BASIC 或是APPLESOFT][中都不可以使用這個敍述；因此，對於程式設計師來說就必須自己設法了。這一節就是告訴您一種解決這

個問題的方法，而它的功用就是能夠顯示出錢號及將錢數算到以分爲單位。

這個副程式就是在模擬 PRINT　USING　的敍述，它能夠處理到 99,999,999.99 的數目，但是它只能顯示出錢號及算到分數。您在進入這個副程式之前必須先定義變數 XD 的值；因此，下面就是呼叫這個副程式的程序；

```
1445 XD = 990542: GOSUB 300: DOL$ = X$
```

產生的結果就放在 X$ 內，而其中就是 990,542.00 。

範例

輸入	結果
XD = 123	X$ = 123.00
XD = 123.4	X$ = 123.40
XD = 123.45	X$ = 123.45
XD = 1234	X$ = 1,234.00
XD = 1234.5	X$ = 1,234.50
XD = 12345	X$ = 12,345.00
XD = 123456.78	X$ = 123,456.78
XD = 1234567.89	X$ = 1,234,567.89
XD = 12345678.99	X$ = 12,345,678.99

```
10 DIM XI(10)
    :
    :

299 REM * PRINT USING
300 FOR JX = 1 TO 10: XI(JX) = 0: NEXT
305 XD = XD * 100: X$ = ""
310 FOR JX = 10 TO 1 STEP -1
315 X1 = INT (XD / 10): X2 = INT (X1 * 10):
     X3 = INT (XD - X2)
```

```
320 XI(JX) = X3: XD = X1
325 NEXT JX
330 IF XI(1) = 0 THEN 340
332 X$= STR$ (XI(1)) + STR$ (XI(2)) + ","
333 X$ = X$ + STR$ (XI(3)) + STR$(XI(4)
     ) + STR$ (XI(5)) + ","
334  X$  =  X$  +  STR$ (XI(6)) + STR$(XI(7)) + STR$
(XI(8)) + "."
335 X$ = X$ + STR$ (XI(9)) + STR$(XI(10))
336 RETURN
340 IF XI(2) = 0 THEN 350
342 X$ = " " + STR$ (XI(2)) + ","
344 GOTO 333
350 IF XI(3) = 0 THEN 360
352 X$ = "     "
354 GOTO 333
360 IF XI(4) = 0 THEN 370
362 X$ = "     " + STR$ (XI(4)) + STR
     $ (XI(5)) + ","
364 GOTO 334
370 IF XI(5) = 0 THEN 380
372 X$ = "      " + STR$ (XI(5)) + ","
374 GOTO 334
380 IF XI(6) = 0 THEN 390
382 X$ = "         "
384 GOTO 334
390 IF XI(7) = 0 THEN 400
392 X$ = "          " + STR$ (XI(7)) +
     STR$ (XI(8)) + "."
394 GOTO 335
400 IF XI(8) = 0 THEN 410
402 X$ = "           " + STR$ (XI(8))
     + "."
404 GOTO 335
410 X$ = "             "
412 GOTO 335
```

圖7-17 PRINT USING程式

7.4.6　過濾數字的副程式

這個副程式可以被用來將一個不是數字的輸入找出：在這個程式中所用的數字可以帶有負號，但是只能位於第一個位置；同時，小數點也只能出現一次。

進入副程式之前，您必須先定義數字在螢光幕上輸入的垂直位置（V）及水平位置（H）。例如；

```
1555 V = 10: H = 20: GOSUB 500
```

當這個程式執行完畢之後，N$ 內存的就是這個數字（以字串型式儲存），如果您想使用這個數字做計算，那麼您可以使用與下面類似的程式行做轉換的工作；

```
1560 N = VAL(N$)
```

```
499 REM * NUMBER FILTER
500 IX = 1: N$ = "": SW = 0
505 VTAB V: HTAB H + IX
510 PRINT "+";
515 VTAB V: HTAB H + IX
520 GET X$
525 IF ASC (X$) = 13 THEN 580:
     REM 13= CARRIAGE RETURN
530 IF ASC (X$) = 8  THEN 560: REM  8= BACKSPACE
532 IF IX > 1 THEN 540
535 IF ASC (X$) = 45 THEN 550
540 IF ASC (X$) = 46 AND SW = 0 THEN
     SW = 1: GOTO 550: REM 46= PERIOD
545 IF ASC (X$) < 48 OR ASC (X$) > 57 THEN 515
550 N$ = N$ + X$: PRINT X$
```

```
555 IX = IX + 1: GOTO 505
560 IF IX = 1 THEN 505
565 HTAB H + IX: PRINT " ";: HTAB H + IX -1
567 IF MID$ (N$,LEN(N$),1) = "." THEN SW = 0
570 N$ = MID$(N$,1,LEN(N$)-1)
575 IX = IX - 1: GOTO 505
580 CALL -868
585 VTAB V: HTAB H + 1: PRINT N$
590 RETURN
```

圖7-18 過濾數字的副程式

7.4.7 過濾字串的副程式

與上面副程式功用相反的就是這個副程式，它將不是文字的輸入找出。

同樣的，在進入這個副程式之前，您必須先定義游標的垂直位置及水平位置。例如：

```
1810 V = 12: H = 8: GOSUB 400
```

最後返回的結果是存在 S$ 內的。

```
399 REM * STRING FILTER
400 IX = 1: S$ = ""
405 VTAB V: HTAB H + IX
410 PRINT "+";
415 VTAB V: HTAB H + IX
420 GET X$
425 IF ASC (X$) = 13 THEN 470
430 IF ASC (X$) = 8  THEN 450
432 IF ASC (X$) = 32 THEN 440
434 IF ASC (X$) = 46 THEN 440
435 IF ASC (X$) < 65 OR ASC(X$) > 90 THEN 415
440 S$ = S$ + X$: PRINT X$
445 IX = IX - 1: GOTO 405
450 IF IX = 1 THEN 405
460 S$ = MID$ (S$,1,LEN(S$)-1)
465 IX = IX -1: GOTO 405
470 CALL -868
475 VTAB V: HTAB H + 1: PRINT S$
480 RETURN
```

圖7-19 過濾字串的副程式

7.5 總 結

在這一章中我們告訴您許多的程式及副程式，這些程式大約可分為排列(sorts)及搜尋(searches) 兩大部份；至於副程式則用來處理輸入的資料並找出不必要的字。

我們同時也將一些數學上的程式告訴給您了，它們相信也能使您知道如何去處理數字了吧！

在這些程式當中，我們同時將程序及程式列出；在大多數的情況下，您還可以看到程式產生的結果。

副程式方面，程式列在書中，並且做了大略的描述，至於呼叫它的方法也詳細地討論過了；每一個副程式都有它獨自的功用，但是您都可以修改它或是將它與您的程式結合起來，使它更符合您的需要。

在本章中還有一節講的是技術與格式，在這一節中我們討論的是如何形成程式、格式的幾個項目。

第8章 作業系統

8.0 大綱

在這一章中我們要討論的是作業系統（ operating systems ）；由於 Apple Ⅱ本身包括了磁帶（ tape ）的作業系統，因此我們將花費一些時間討論它；但是，本章討論的重點是放在功能較大的磁碟作業系統（ DOS ）上的。

由於 Apple 的 DOS 大約是在四年前所發展出來的，因此到現在爲止已經經過了許多的改變。第一套的版本是 3.1，是在 1980 年的 7 月發展出來的；第二個重要的版本則是 DOS 3.2 ，在 1979 年的二月發展出來的；其後，在 1979 年的 7 月，又提供了 3.2.1，而這套作業系統對於 3.1 而言是一個很大的進步，比方說， DOS 3.1 只能讀大寫的字母，而這套版本則加入了許多新的程式，例如 UPDATE 3.2 及 MASTER CREATE ，同時您可以多使用兩種型式的檔— S 及 R（但是並不能用命令來使用它們)，並且可以加入自動啓動的 ROM （ AUTOSTART ROM），而當發生 DISK FULL ERROR 的錯誤時，它會自動地將所有被打開的檔關閉。

目前 Apple][的磁碟機所使用的作業系統是 DOS 3.3 ，它具有的功能更大，而磁碟的結構是以 16 磁區（ sector ）爲單位的，同時 FID 及 MUFFIN 程式也被加入了， COPY 程式則較以前更爲精良，而且在您 SAVE 一個程式時， DOS 會自動地執行 VERIFY 命令。

在本章所有的討論中，我們將專注於 DOS 3.3，這是因爲 DOS 3.3

是目前最有展望的一套作業系統。

在第二章中，我們曾經對您介紹過了電腦硬體上的安排情形，現在，我們則使用另外一種觀點來看電腦，我們將以邏輯的安排方式介紹給您。圖8－1就是一個電腦的邏輯安排圖。

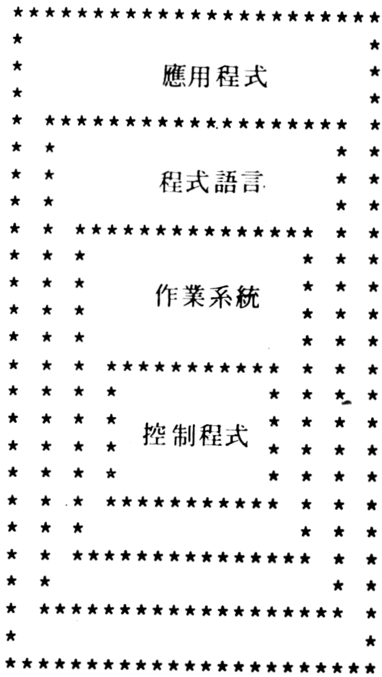


圖8－1 電腦的邏輯安排

我們可以由於圖中知道控制程式（ Control Program ）就是整個系統的核心，而這也就是 Apple][內的監督系統（ monitor ）， 是處於整個系統的最低層次；在這個層次中，只有一些最基本的功能會被執行，像是"boot" PROM的工作可能是它唯一能夠執行的功能，但是對於 Apple][而言，它則可以執行以下的許多功能；像是：

* —字的輸入及輸出
    * —由鍵盤接受字的輸入
    * —將輸入的字顯示在螢光幕上。
* —低解析度的繪圖
    * —繪出一個點
    * —繪出一條線
* —由監督系統使用發聲器（ Speaker ）
* —遊戲控制器，按扭及電鈴（ annunciator ）的輸入及輸出
    * —感應遊戲控制器的位置
    * —感應按扭是否被按下。
    * —感應電鈴是否被使用
* —磁帶機的輸出與輸入
    * —由磁帶將程式存入記憶體
    * —由記憶體將程式存入磁帶
* —螢光幕的命令
    * —消除螢光幕
    * —消除某一行
    * —消除某一段螢幕
    * —設定往上捲的範圍大小
    * —設定大字資料情況（ text mode ）
    * —設定圖形情況（graphics mode ）
* —自動啓動的功能
* —RESET機器

以上所列的功能並不詳細，它只是告訴您監督系統所能執行的功能的型式。監督系統所能執行的許多功能對於整個機器來說是非常重要的。

圖 8 — 1　假設的作業系統是以磁碟爲基礎的；至於磁帶系統則是較簡單的系統，我們將在下一節討論它。任何作業系統的目的就是操作整個的電腦系統；一般來說，它都是執行像是存入語言，存入程式，儲存檔／取用檔等等的工作；而作業系統實際上就是一個軟體程式，它將整個電腦硬體的功

能發揮到極至。

只要您將機器打開，或是 boot 您所要的作業系統，那麼它就會被存入記憶體內，執行它的功能了。在這個時候，就拿 Apple][來說，您就可以使用一種語言操作它了。

這個語言將允許您（經過作業系統） LOAD 或 RUN 一個應用程式；我們對於任何程式，都將加上"應用"兩個字，這是因爲您使用這個程式的目的，必定是希望它幫助您解決一些問題，像是一般帳之類的。

8.1　磁帶系統 (Cassette System)

磁帶的作業系統只有兩個 BASIC 命令，那就是 SAVE 及 LOAD ，這兩個命令都不需要使用檔的名字。

我們假設設置磁帶機的工作已經完成了；我們建議您參考 Apple][的手册研究如何設置磁帶機。

一個整數 BASIC 或是 Applesoft][的程式是以記錄（record) 的型式存在磁帶上的；一個記錄也就是任何數量的資料以一定的格式存在某種設備內。程式是以兩個記錄的方式儲存的，第一個記錄內存的是程式的長度，而第二個記錄內存的則是程式本身。圖 8 － 2 就是這些記錄的格式。

```
+--------+---+----+---+---------+---+----------+---+
!   頭   ! S ! LB ! C !    頭   ! S !   程式   ! C !
+--------+---+----+---+---------+---+----------+---+

        注意： S = Sync. 數元
               C = 檢查總數數元組
              LB = BASIC 程式的長度
```

圖 8 - 2　磁帶的格式

"頭"內存的是 10 秒，770Hz 的言調，這個"頭"的功用就是要使磁帶機的馬達能夠達到固定的速度，並且讓磁帶的前頭部份空下來不被使用。

至於 sync 數元則用來代表"頭"的結束及資料的開始。

資料在磁帶上的儲存位置是由低位開始而到高位結束的；而當資料由電腦中移出時，則是依照MSB(most significant byte)先而 LSB (least significant byte)最後的次序存入磁帶內的，移出的速度則是 1500 bps(bit per second)

檢查總數(checksum)是一個儲存位置，它用來計算資料是否有效；這個數元組當資料被寫入磁帶內的動作結束後被加在資料區的後面；如果當您想將存在磁帶上的程式讀出的時候，讀的動作產生的檢查總數。數之組與存在磁帶上的檢查總數的值相同，那麼您就可以認為這個程式是按照原先的情況被存回記憶體內的了。

8.2 磁帶系統

學習使用 DOS 的過程最主要的就是瞭解 DOS 執行的功能，然後學習如何將它們加入到 Applesoft][的程式中。接下去要介紹的大多數的指令及程式都是使用 Applesoft][語言設計的，我們將在適當的時機告訴您如何在 Applesoft][內使用 DOS 的命令，這使您能夠將它們結合為一體，視 DOS 的命令為 Applesoft][語言的一部份。

接下去的幾節將討論 DOS 的命令，兩種主要的文字資料檔的結構，二進位檔(binary file)以及 EXEC 命令，相信在您研究完畢之後，您已經能夠有效地運用 Dos 了。

8.2.1 一般地考慮

為了要使 DOS 能夠在 Apple][內被使用，您首先必須 "boot" 磁片；下面兩種方法可以使您達成這個目的。

首先，您可以將 System master 磁片放入磁碟機內，並且將機器的開關打開；這時候只要您的機器具有自動啓動的 ROM ，那麼您就已經完成了 boot 的工作了。

第二種方法就是將一片磁片放入磁碟機內，打開機器的開關，然後使用

下面的 BASIC 命令 "boot" DOS。

PR#s 例如]PR#6

或 IN#s 例如]IN#6

S＝磁碟介面卡位於的擴充接點數目

只要您將作業系統存入記憶體內之後，您就可以使用它了；您最先要做的工作就是準備一片空白的磁片用來儲存程式或是檔，而這個準備的工作我們就稱它當設定格式（ initializing ）。

8.2.1.1 設定磁片的格式

一片空白的磁片（剛剛由製造商製成的）內並不具有必須的訊息可以供您使用 DOS 將您的資料存入磁片內，您必須將磁片的格式設定，把(VTOC)（Volume Table of Contents），磁軌（track）及磁區的標識，以及 DOS 存入磁片內才可以。下面就是設定磁片格式的過程：

注意： 1－將 system master 磁片插入第一個磁碟機內。

2－做 Boot 的動作。

3－打入下面的命令。

]LOAD HELLO（由 system master ）

注意： 1－將 system master 移開。

2－將空白的磁片插入第一個磁碟機內。

3－打入下面的命令。

]INIT HELLO

圖 8－3 設定磁片的格式

在您按下了 RETURN 鍵之後，第一個磁碟機上的 "IN USE" 燈就會亮著，而您也可以聽到磁碟機轉動的聲音；當 DOS 將磁片的格式設定好了之後，

它會將 DOS 3.3 複製一份到第0,1及第2個磁軌上去。

在您做完了這個工作之後， HELLO 程式就已經存在磁片上，現在您也就可以將您的程式或檔存入磁片內了。

8.2.1.2 SAVE及LOAD程式

如果您有整數 BASIC 或是 APPLESOFT][的程式存在記憶體內，您都可以將它們存到磁片上去。您執行下面的命令。

]SAVE 檔的名稱　　　　例如]SAVE DUMMY

這時候 "IN USE" 的燈會亮著，磁碟機則開始轉動；當這個程式被寫入磁片內時，您將會聽到刺耳的摩擦聲。

現在我們要討論一個非常重要但是經常被忽視的問題。 DOS 3.3 與其他的磁碟作業系統相同，都是不具毀壞性的；這就是說，不論您是要SAVE或是 LOAD 一個程式，這個程式的原始資料並不會被毀壞。當您將一個程式存入磁片內時，存入的程式只是記憶體內程式的複製而已，而原先在記憶體內的程式仍然存在；至於您由磁碟 LOAD 一個程式時，也是同樣的情況，磁片內的原始資料並不會消失。

我們可以明顯地看出來，當您想要將一個程式存到磁片上時，您必須將它存入記憶體內，而每一個存在磁片上的程式都必須具有一個獨一的名字。在您想將這個程式存入記憶體時，您可以使用下面的命令：

]LOAD 檔的名稱　　　　例如：]LOAD DUMMY

現在我們已經知道如何 SAVE 及 LOAD 一個程式了，讓我們設計一個小的程式，並且將它 SAVE 到磁片上，再由磁片上 LOAD 到記憶體內，打入下面立即執行的命令：

]NEW

注意：這個命令將記憶體內現存的程式消除掉。

```
1 REM ***** MATH DEMO *****
2 REM *
3 REM * WRITTEN BY:JL CAMPBELL
4 REM *       DATED:07/12/81
5 REM *
6 REM * LANGUAGE:APPLESOFT II
7 REM * COMPUTER:APPLE II PLUS
8 REM *
9 REM *********************
10 TEXT: CLEAR: POKE 216,0: A = FRE(0)
20  HOME: PRINT: PRINT TAB( 11)"*** MATH  DEMO ***":
PRINT
30 FOR I = 1 TO 10
40 PRINT I, INT ( I * 2), INT (I ^ 2)
50 NEXT I
60  VTAB  20:  CALL  -958:  INPUT  "HOW  ABOUT  ONCE
MORE?(Y/N) ";A$
70 IF LEFT$ (A$,1) = "Y" THEN 20
80 IF LEFT$ (A$,1) = "N" THEN 100
90 VTAB 20: CALL -958: HTAB 10
92 PRINT "ILLEGAL ENTRY!";CHR$(7)
94 FOR J = 1 TO 1000: NEXT J: GOTO 60
100 VTAB 4: CALL -958: VTAB 12
110 PRINT "GOODBYE FOR NOW....": END
```

圖8-4　MATH DEMO程式

```
]RUN

              *** MATH DEMO ***

1                  2                  1
2                  4                  4
3                  6                  9
4                  8                  16
5                  10                 25
6                  12                 36
7                  14                 49
8                  16                 64
9                  18                 81
10                 20                 100
```

現在我們已經設計好了這個程式，並且執行了它證明一切工作都很順利。我們可以將它存入磁片上了。

```
]SAVE MATH DEMO,D1

]NEW

]LOAD MATH DEMO
```

我們已經將這個程式以 MATH DEMO 的名稱存入磁片上了，並且將記憶體內原先的程式清除乾淨，再將磁片上的這個程式又存回記憶體內了，如果您現在打入 LIST 命令，您將看到這個完整的程式又存回到記憶體內了，同時在磁碟片上仍然保存著這個程式的複製。打入 CATALOG 命令，您將看到:

```
DISK VOLUME 254

 A 002 HELLO
 A 003 MATH DEMO
```

8.2.1.3 RENAME及DELETE命令

有時候您也許希望改變一個磁碟檔的名字；假設您現在想將 MATH DEMO以爲DUMMY，那麼您可以使用下面的命令：

```
]RENAME MATH DEMO,DUMMY
```

這時候磁碟機上的 "IN USE" 燈會亮著，磁碟機轉動一會之後，就把這個新的名字寫入了。

特別注意： DOS 3.3 並不檢查這個新的名字是否代表著其他的磁碟檔， DOS 只是將新的名字代替舊的名字。

同樣的，有時候您想要 DELETE 一個程式或檔，那麼您可以使用下面

的命令：

```
]DELETE GENERAL LEDGER
```

請在您使用 DELETE 命令清除一個磁碟檔之前，仔細地想一想您是否確定要消除這個程式或資料，因為在檔被消除之後就幾乎沒有可能再重新得回了。

8.2.1.4 CATALOG命令

CATALOG命令被使用來觀看磁碟上儲存了什麼東西；當您打入CATALOG 命令並按下了 RETURN 鍵，下面的情形將發生：

* — "IN USE" 的燈會亮著。
* —磁碟機將轉動一會。
* —磁片的目錄將被顯示出來。

試著打入這個命令，您將看到

```
]CATALOG

DISK VOLUME 254

 A 002 HELLO
 A 003 DUMMY
```

注意；如果磁片內存有整數 BASIC 的程式，那麼在檔的前面是 I 字而非 A 字。

我們由以上的目錄看出這片磁片的磁片號碼(volume number)是254-設定值，這兩個存在磁片上的檔是 APPLESOFT][的程式： HELLO 程式佔用了兩個磁區，而 DUMMY 佔用了 3 個磁區。

一個磁區可以儲存 256 個數元組的資料；因此，您可以很快地估計出程式的長度；但是這只是一個估計的數字，因為磁區內存的還有一些像是檔名

稱，磁軌及磁區的利用情形等等的資料，它們也佔用儲存位置。

8.2.1.5 LOCK及UNLOCK命令

您可以將一個程式或資料檔鎖住(LOCK)，這就表示這個檔只能被讀出資料而不能被寫入其他的資料。這對保護您的資料而言非常有用。同樣地，您也可以將一個被鎖住的檔打開，使用的命令就是 UNLOCK 。所有的檔被存入磁碟內時，都是以打開的型式存入的，您對重要而不想被毀壞的檔就必須使用 LOCK 命令將它們鎖起來。現在讓我們將我們磁片上的兩個檔鎖起來吧：

```
]LOCK HELLO
]LOCK DUMMY
```

再讓我們看看磁片的目錄：

```
]CATALOG

DISK VOLUME 254

*A 002 HELLO
*A 003 DUMMY
```

您可以發現在這兩個被鎖住的檔的前面有一個星號，現在它們都是可讀不可寫了。

8.2.1.6 FP及INT指令

Apple][具有同時將整數 BASIC 與 APPLESOFT][存入供您使用的能力；您可以在 DOS 3.3 的控制下轉換著使用這兩種語言；下面就是這兩個命令：

```
>FP     注意：轉往APPLESOFT ][ 語言。
]INT    注意：轉往整數 BASIC 語言。
```

任何存在記憶體內的程式都被清除掉了； DOS 3.3 或是您轉換過來的 BASIC 語言都不會將您的程式清除，但是由於指標，及程式間的連接都改變了，因此，事實上您的程式已經不存在了。

8.2.1.7 磁碟的格式

DOS 3.3 使用 Disk][磁碟機時，它將磁片分為 35 條磁軌，每條磁軌則有 16 個磁區。最外圍的磁軌編號為零，而最內層則為 34 。磁碟機的讀寫頭當磁片旋轉的時候會往前或往後移動而讀或寫任何磁軌或磁區上的資料。

磁軌上每一個磁區可以儲存 256 個數元組的資料；為了要將資料存入磁片內， DOS 必須首先將 256 個數元組的資料存入到一個檔的緩衝區(bu-ffer)內，然後， DOS 才能將這 256 個數元組的資料寫入磁片內。至於緩衝區就是記憶體內的一塊區域，它只是暫時地被用來做為儲存的位置。

DOS 3.3 對於磁軌及磁區的使用情形都記錄地非常清楚；更進一步地說， DOS 3.3 會去找出可供使用的儲存位置。所有磁軌及磁區的使用情形都被記錄在第 17 個磁軌的索引檔(directory)內。

至於上面討論的磁碟命令您可以參看附錄D或是 Apple 公司的DOS 手册。

下面的表就是磁片的邏輯圖。

```
                          磁軌

                  1         2         3
        01234567890123456789012345678901234
       0                 V
       1
       2
       3DDD              C
       4OOO              A
   S   5SSS              T
   E   6                 A
```

```
C   7III              L
T   8MMM              O
O   9AAA              G
R  10GGG
S   1EEE               HD
    2                  EU
    3                  LM
    4                  LM
    5                  OY
```

表 8-1 典型的磁碟圖

如您所見到的，DOS 是存在第 0 到第 2 個磁軌內的；至於磁片的目錄則與 VTOC，V 同時存在第 17 個磁軌內。

當系統 "boot" 的時候，DOS 的複製就被讀入記憶體內了，並且由 HIMEM 的位置開始存放，用時也被保護起來。只要 "boot" 的動作完成了，您的系統就可以執行或是存入應用程式了。注意 HELLO 及DUMMY 並不是存在相同的磁軌內的，但它們都是由第 15 個磁區開始存放；這是當磁片上有足夠的位置供資料存放時的正常儲存情形。當一個程式被設定格式之後，"boot" 程式的名稱一在這裏是 HELLO ，就與 DOS 3.3 存放在一起，同時也將起始位置，指標存到 VTOC 內。VTOC 一般都被存在第 17 個磁軌的第 0 個磁區內，至於檔（程式，文字資料檔，二進位檔等等），則可以存放在磁片的任何地方。VTOC 內存的是：

* 一第一個存有目錄磁區的磁軌數目。
* 一第一個被使用的磁區的磁區數目。
* 一磁碟片號碼。
* 一最後被存入資料的磁區的磁軌數目。
* 一磁軌使用的情形。
* 一磁片上磁軌的數目。
* 一磁軌內磁區的數目。
* 一每一個磁區內數元組的數目。
* 一對每一個磁軌而言，其內空白磁區的數元圖（Bit map）。

＊一對於未來增加的磁軌的數元圖。

DOS 3.3 到目錄內找到第一個檔的方法是讀取 VTOC 而找到存有第一個目錄的磁軌及磁區的數目，正常地情況之下，這些都是存在第 17 個磁軌的磁區內的。第一個磁軌／磁區的指標是指到第 17 個磁軌的第 15 個磁區的，而存在第 15 個磁區內的磁軌／磁區的指標則指到第 17個磁軌內的第 14 個磁區，這種情況一直繼續下去直到磁區指標內指的值是零爲止，而這也就表示目錄的結束，像這樣的磁軌／磁區的安排方式就稱它爲連接 (link) 表。在最簡單情況下的連接就是在表中的每個項目內都存著指到下一個項目的指標，而在表中的最後一個項目內則存著表示結束的註記。

在第 17 個磁軌內的每一個磁區內都存有一個磁軌／磁區表(track/ector list)，其中包含著：

＊一下一個存有目錄磁區的磁軌數目。

＊一下一個存有目錄的磁區數目。

＊一檔的描述。

檔的描述內存有相關於這個檔的重要訊息，像是檔的型式，檔的名字，檔的長度，儲存的位置等等；在一個磁軌磁區表內最多可以描述 7 個檔。

我們已經大略地描述過了 DOS 3.3 尋找磁片內資料的方法了，經由 DOS 的幫助您就可以 READ, WRITE, LOAD 及 SAVE 任何檔了。

下面的表告訴您磁片的組織情形，依照各種不同型式的 DOS 作爲描述的的分類。

磁軌——所有的型式	35
磁區內的數元組數—所有的型式	256
磁軌內的磁區數—DOS 3.3	16
磁片內的磁區數—DOS 3.3	560
可被使用的磁區數— DOS 3.3	496

磁片內的數元組數－DOS 3.3	143360
磁片內可用的數元組數－DOS 3.3	126976
磁軌內的磁區數－ DOS 3.2.1 及較早的型式	13
磁片內的磁區數－ DOS 3.2.1 及較早的型式	455
可被使用的磁區數－ DOS 3.2.1 及較早的型式	403
磁片內的數元組數－ DOS 3.2.1 及較早的型式	116480
磁片內可用的數元組數－DOS 3.2.1 及較早的型式	103168

表 8 － 2 磁片的組織

8.2.2 檔的結構

有許多時候您希望將不是程式的訊息或資料存到磁片上去；例如，庫存的資料，應收帳及地址等等。 DOS 3.3 允許您做這個工作。

執行這項工作的方法就是您設計一個程式將這個檔建在磁片上，而這個檔內存的就是您想要保留的訊息；然後，您在使用這些資料的時候又必須另外設計一個程式來找尋這些資料供使用者使用；這個一連串的步驟實際上包含了三個程式，但是可以放在一個程式內,當做副程式使用，事實上，對於一個檔的處理工作只包含了七個步驟；它們分別是：

建立的程式

* －建檔
* －消除檔

處理的程式

* －輸入資料
* －改變資料
* －消除資料

輸出的程式

* －查看檔內的資料

* 一印出報告

圖8 -5 處理檔的程式

一個檔就是一列的資料依照一定的次序排列，而這個一系列的資料可以是任何的大小，甚至大到磁片的極限都可以，其中可以存著任何訊息，甚至是一個程式也可以，檔可以在 DOS 3.3 的控制之下建立成文字資料檔，稱它爲文字資料檔的原因是資料是以文字資料的型式存在檔內的；DOS 3.3 本身可以建立兩種型式的文字資料檔－順序處理檔及隨機處理檔（sequential and random access files）

這兩種檔的資料儲存方式並沒有不同，但是在檔的特性方面卻是不同的。

T字就是代表文字資料檔；因此，我們現在已經有了三種表示檔的代表字母了。它們分別是：

* 一A，表示Applesoft][程式
* 一I，表示整數 BASIC 程式
* 一T，表示文字資料檔(Text file）

由於文字資料檔並不是程式而是資料，因此您不能夠 RUN，SAVE 或 LOAD 它；您必須使用其他的命令去處理它。

8.2.2.1 順序處理檔

順序處理檔（sequential file）是一個線性排列的資料列，裏面包含了文字或是數字，而其中的每一個項目我們可以視它爲檔中的一個記錄（record）；這種檔的型式是在以前只有磁帶機的時代裏發展出來的；因爲磁帶機具有的特性就是您必須由一個檔的起頭開始讀取每一個記錄，並且執行處理的工作；而一個存在磁片上的順序處理檔也具有同樣的特性，不論您是讀或寫資料都必須依照順序完成。

爲了要使我們能夠建立檔，將資料存入這個檔內，以及由這個檔內取出資料，我們都必須先建立程式與 DOS 間的聯繫；等到這個聯繫建立完成之

後，我們才可以告訴 DOS 3.3 我們是要讀或寫資料；然後APPLESOFT]的 BASIC 程式才能夠通知 DOS 3.3 我們所要採取的處理方式；這時候我們就可以輸入資料或是印出資料了；等到聯繫的工作完成了之後，我們必須做結束這個聯繫的工作。

我們現在設計一個程式能夠儲存及取用一個四年級班級的學生名字，我們必須能夠在螢光幕上觀看這些資料，印出這個班級所有人員名字的報告，以及改變或消除班級成員的名字。

```
1   REM *** CLASS LIST ***
2   REM *
3   REM * WRITTEN BY:JL CAMPBELL
4   REM *        DATED:07/16/81
5   REM *
6   REM *   LANGUAGE:APPLESOFT
7   REM *   COMPUTER:APPLE II
8   REM *
9   REM ****************
10   TEXT : CLEAR : POKE 222,0:A =  FRE (0)
20   DIM CL$(50)
45 D$ =  CHR$ (4)
50   GOTO 800
51   REM **************
52   REM * GEN'L SUBR *
53   REM **************
55   INPUT "WHICH ONE? ==> ";N: RETURN
60   VTAB (22): CALL  - 958: HTAB 13: INVERSE : PRINT
"ILLEGAL ENTRY!"; CHR$ (7): NORMAL
64   FOR L = 1 TO 1000: NEXT : RETURN
70   VTAB 4: CALL  - 958:IX = R
75   PRINT "THERE ARE ";R;" STUDENTS IN ";F$: RETURN
100 SW = 0
105  VTAB 20: INPUT "ANOTHER NAME?(Y/N) = ";A$
110  IF  LEFT$ (A$,1) = "Y" THEN SW = 1: RETURN
115  IF  LEFT$ (A$,1) = "N" THEN SW = 0: RETURN
120  GOSUB 60: GOTO 100
800  HOME : PRINT : PRINT  TAB( 12)"*** FILE DATA ***":
PRINT
802  PRINT "TYPE 'RETURN' KEY IF FILE NOT CREATED!"
805  VTAB 6: CALL  - 958
810  INPUT "ENTER FILE NAME = ";F$
815 L =  LEN (F$): IF L < 2 THEN 900
820  IF L > 20 THEN F$ =  LEFT$ (F$,20)
```

```
825 F$ = "CLASS." + F$
830  ONERR  GOTO 8000
835  PRINT D$;"OPEN ";F$;",D1"
840  PRINT D$;"READ ";F$
845  INPUT R
850  IF R < 1 THEN 870
855  FOR I = 1 TO R
860  INPUT CL$(I)
865  NEXT I
870  PRINT D$;"CLOSE"
875  POKE 216,0

899  REM * MAIN DRIVER
900  HOME : PRINT : PRINT  TAB( 9)"*** CLASS LIST PROG
***": PRINT
905  PRINT "MAIN MENU:": PRINT "---- ----"
910  PRINT  TAB( 10)"1- CREATE CLASS FILE"
912  PRINT  TAB( 10)"2- DELETE CLASS FILE": PRINT
914  PRINT  TAB( 10)"3- ENTERS NAMES"
916  PRINT  TAB( 10)"4- CHANGE NAMES"
918  PRINT  TAB( 10)"5- DELETE NAMES": PRINT
920  PRINT  TAB( 10)"6- REVIEW NAMES"
922  PRINT  TAB( 10)"7- PRINTS NAMES"
928  PRINT : PRINT  TAB( 10)"0- EXIT PROGRAM"
930  VTAB (18): CALL  - 958: GOSUB 55:N =  INT (N)
935  IF N = 0 THEN 990
940  IF N > 0 AND N < 8 THEN 950
945  GOSUB 60: GOTO 930
950  ON N GOSUB 1000,2000,3000,4000,5000,6000,7000
960  GOTO 900
990  HOME : CALL  - 958: VTAB 10: PRINT "GOODBYE FOR
NOW...."
995  END
1000  HOME : PRINT : PRINT  TAB( 9)"*** CREATE FILE
***": PRINT
1005  ONERR GOTO 8000
1010  PRINT "NAME LESS THAN 20 CHARS."
1020  VTAB 6: CALL  - 958
1030  INPUT "ENTER FILE NAME = ";F$
1040 L =  LEN (F$): IF L < 2 THEN 1999
1050  IF L > 20 THEN F$ =  LEFT$ (F$,20)
1060 F$ = "CLASS." + F$:N = 0
1100  PRINT D$;"OPEN ";F$
1110  PRINT D$;"DELETE ";F$
1120  PRINT D$;"OPEN ";F$
```

```
1130  PRINT D$;"WRITE ";F$
1140  PRINT N
1150  PRINT D$;"CLOSE"
1160  POKE 216,0
1999  RETURN
2000  HOME : PRINT : PRINT  TAB( 9)"*** DELETE FILE
***": PRINT
2100  PRINT D$;"DELETE ";F$
2999  RETURN
3000  HOME : PRINT : PRINT  TAB( 10)"*** ENTERS NAMES
***": PRINT
3010  GOSUB 70
3200 IX = R
3210  VTAB 6: CALL  - 958: VTAB 10
3220 IX = IX + 1:R = IX
3230  PRINT "ENTER NAME #";IX;: INPUT " = ";CL$(IX)
3240  GOSUB 100
3250  IF SW = 1 THEN 3000
3280  PRINT D$;"OPEN ";F$
3290  PRINT D$;"DELETE ";F$
3300  PRINT D$;"OPEN ";F$
3310  PRINT D$;"WRITE ";F$
3320  PRINT IX
3325  IF IX = 0 THEN 3360
3330  FOR I = 1 TO IX
3340  PRINT CL$(I)
3350  NEXT I
3360  PRINT D$;"CLOSE ";F$
3370 R = IX
3999  RETURN
4000  HOME : PRINT : PRINT  TAB( 10)"*** CHANGE NAMES
***": PRINT
4010  GOSUB 70
4220  VTAB 6: CALL  - 958
4230  INPUT "WHICH STUDENT #? = ";RN
4240  VTAB 8: PRINT "STUDENT #";RN;" = ";CL$(RN)
4250  VTAB 10: INPUT "CHANGE TO WHAT? = ";CL$(RN)
4260  GOSUB 100
4270  IF SW = 1 THEN 4000
4300  GOTO 3280
5000  HOME : PRINT : PRINT  TAB( 10)"*** DELETE NAMES
***": PRINT
5010  GOSUB 70
5120  VTAB 6: CALL  - 958
```

```
5130  INPUT "WHICH STUDENT #? = ";RN
5140  VTAB 8: PRINT "STUDENT #";RN;" = ";CL$(RN)
5150  VTAB 10: PRINT "DELETE STUDENT ";CL$(RN);: INPUT
"?(Y/N) ";A$
5160  IF  LEFT$ (A$,1) = "Y" THEN 5200
5170  IF  LEFT$ (A$,1) = "N" THEN 5000
5180  GOSUB 60: GOTO 5150
5200  FOR I = RN + 1 TO R
5210 CL$(I - 1) = CI$(I)
5220  NEXT I
5225 CL$(R) = "   "
5230 R = R - 1:IX = R
5240  IF R = 0 THEN 5500
5250  GOSUB 100
5260  IF SW = 1 THEN 5000
5270  GOTO 3280
5500  VTAB 4: CALL  - 958
5510  PRINT "THERE ARE ZERO STUDENTS STORED."
5520  PRINT "RECREATING THE FILE TO REFLECT THIS!"
5530  PRINT : PRINT "PLEASE WAIT A MOMENT."
5540  GOTO 3280
5999  RETURN
6000  HOME : PRINT : PRINT  TAB( 10)"*** REVIEW NAMES
***": PRINT
6010  FOR I = 1 TO R STEP 10
6020  VTAB 4: CALL  - 958
6030  FOR J = I TO I + 9
6040  IF  LEFT$ (CL$(J),1) = "" OR  LEFT$ (CL$(J),1) =
" " THEN 6060
6050  PRINT J;"- ";: HTAB 5: PRINT CL$(J)
6060  NEXT J
6070  VTAB 18: INPUT "TYPE 'RETURN' KEY TO CONTINUE...
";A$
6080  NEXT I
6090  RETURN
7000  HOME : PRINT : PRINT  TAB( 10)"*** PRINTS NAMES
***": PRINT
7010  PRINT "DO THE FOLLOWING:"
7020  PRINT  TAB( 5)"1- LOAD PRINTER PAPER"
7030  PRINT  TAB( 5)"2- TURN ON PRINTER"
7040  PRINT  TAB( 5)"3- SET TOP-OF-FORM"
7050  PRINT  TAB( 5)"4- PRINTER ON-LINE"
7060  PRINT : INPUT "TYPE 'RETURN' KEY TO CONTINUE...
";A$
7070  VTAB 12: CALL  - 958
```

```
7080  INPUT "ENTER PRINTER SLOT(1-7) = ";SL:SL =  INT
(SL)
7090  IF SL > 0 AND SL < 8 THEN 7100
7095  GOSUB 60: GOTO 7070
7100  VTAB 14: CALL  - 958
7110  PRINT "PRINTER TYPE:"
7120  PRINT  TAB( 5)"1- SERIAL"
7130  PRINT  TAB( 5)"2- PARALLEL"
7140  PRINT : GOSUB 55:SP =  INT (N)
7150  IF SP = 1 OR SP = 2 THEN 7160
7155  GOSUB 60: GOTO 7100
7160  PR# SL
7170  IF SP = 2 THEN  PRINT  CHR$ (9);"80N"
7180  PRINT  TAB( 28)"*** CLASS LIST REPORT ***": PRII
" "
7190  PRINT  TAB( 10)"CLASS LIST FOR ";F$;" FILE."
7200  PRINT " "
7210  PRINT  TAB( 11)"#";: POKE 36,20: PRINT "NAME"
7220  PRINT  TAB( 10)"---";: POKE 36,20: PRINT "----"
7230  PRINT " "
7240  FOR I = 1 TO R
7250  PRINT  TAB( 10)I;: POKE 36,13: PRINT "-";
7260  POKE 36,20: PRINT CL$(I)
7270  NEXT I
7280  PRINT  CHR$ (12)
7290  IF SP = 2 THEN  PRINT  CHR$ (9);"40N"
7300  PR# 0
7990  RETURN

7999  REM * ERROR ROUTINES
8000 E =  PEEK (222): VTAB 20: CALL  - 958
8010  PRINT "ERROR = <";E;"> ";
8020  POKE 222,0 : CLEAR
8030  ON E GOTO 8100,8150,8150,8200,8250,
       8300,8100,8100,8350,8400
8100  PRINT "CONSULT MANUAL"
8110  GOSUB 64: GOTO 800
8150  PRINT "RANGE ERROR"
8160  GOSUB 64: GOTO 800
8200  PRINT "WRITE PROTECTED"
8210  VTAB 22: CALL  - 958: PRINT "REMOVE PROTECTION.
TYPE 'RETURN' KEY."
```

```
8220  GET K$: IF  ASC (K$) = 13 THEN  PRINT : GOTO 800
8230  GOSUB 60: GOTO 8210
8250  PRINT "FILE NOT CREATED"
8260  PRINT D$;"DELETE ";F$
8270  GOSUB 64: GOTO 900
8300  PRINT "FILE NOT FOUND"
8310  GOSUB 64: GOTO 800
8350  PRINT "DISK FULL"
8360  VTAB 22: CALL  - 958: PRINT "CHANGE DISKETTES.
TYPE 'RETURN' KEY."
8370  GET K$: IF  ASC (K$) = 13 THEN  PRINT : GOTO 800
8380  GOSUB 60: GOTO 8360
8400  PRINT "FILE LOCKED"
8410  VTAB 22: CALL  - 958: PRINT "UNLOCK CLASS FILE."
8420  FOR I = 1 TO 10: PRINT : NEXT I
8430  END
```

圖8-6 **CLASS LIST程式**

```
]RUN CLASS LIST

         *** CLASS LIST PROG ***

MAIN MENU:
---- ----
            1- CREATE CLASS FILE
            2- DELETE CLASS FILE

            3- ENTERS NAMES
            4- CHANGE NAMES
            5- DELETE NAMES

            6- REVIEW NAMES
            7- PRINTS NAMES

            0- EXIT PROGRAM

WHICH ONE? = 7
```

圖8-7 **CLASS LIST程式的主手册**

```
          *** PRINTS NAMES ***

CLASS LIST FOR CLASS.4TH GRADE FILE.

 #            NAME
---           ----

1  -          JOHN CAMPBELL
2  -          DEBBIE WALTERS
3  -          JOE JACKSON
4  -          GEORGIA SMITHE
5  -          LANCE ZIMMERMAN
6  -          DORIS SCOTT
7  -          PAUL DUKE III
8  -          GAIL JOHNSON
9  -          TONY MARSHALL
10 -          FRED GATES
11 -          MARSHA TUCKER
12 -          BARBARA TUCKER
13 -          J. Q. PUBLIC III
14 -          MARK RAYMOND
15 -          WILLIAM SPOCK
```

圖 8-8 CLASS LIST的報表

```
15
JOHN CAMPBELL
DEBBIE WALTERS
JOE JACKSON
GEORGIA SMITHE
LANCE ZIMMERMAN
DORIS SCOTT
PAUL DUKE III
GAIL JOHNSON
TONY MARSHALL
FRED GATES
MARSHA TUCKER
BARBARA TUCKER
J. Q. PUBLIC III
MARK RAYMOND
WILLIAM SPOCK
```

圖 8-9 4TH GRADE檔

圖8－6就是我們所建立的程式，我們採取圖4－8中的架構加上了幾個副程式而完成了這個程式；這個程式的架構及邏輯我們已經討論過了；同時我們也可以將4－8中的程式由磁碟存入記憶體內，加以修改，再用目前的名字存回磁碟。這個程式中有七個主要的副程式，也就是在圖8－5中所列七個項目；主要的手册在圖8－7中顯現給您了；至於產生報告的副程式則只會印出班上成員名單的一份報表，這在圖8－8中也顯示出來了；而這個順序處理的文字資料檔－名字叫做 CLASS.4TH GRADE －就在圖8－9中，它裏面存的就是學生的總數以及學生的名字。

這個程式非常地直接一點也不曲折。

行號1－50－程式的起頭；設定行列，清除機器等。

行號50－799 －一行的副程式及中型大小的副程式。

行號800－875－要求輸入檔的名稱，如果檔還沒有被建立，那麼您按下 RETURN 鍵；如果您輸入的是正確的檔的名字，那麼這個檔的內容就被讀入記憶體內。

行號899－995－主要的手册在這裏顯示供使用者選擇。這是程式中唯一的出口。

行號1000－1999－這一段建立一個新檔。

行號2000－2999－消除檔。

行號3000－3999－允許您輸入新的名字。

行號4000－4999－允許您更改學生的名字。

行號5000－5999－允許您消除名字。

行號6000－6999－允許您由螢幕上一次看10個名字。

行號7000－7999－印出班級學生名字的報表。

現在讓我們看看 DOS 3.3 的命令；第一組要看的是1060－1150的程式行；檔的名字存在 F$ 內，由於您輸入的檔名字的第一個可能是數字，因此我們將 CLASS. 連接在前面；這是因爲檔的名字的第一個字必須是字母，所以我們才這麼做。

位於1100及1110行號的敍述建立一個新的檔，這個工作設立了 VTOC

(目錄)內檔的資料，而 DELETE 命令將這個尙未有任何資料的檔消除掉；因此，目前我們已經可以 OPEN 這個檔用來儲存實際的資料了。

接著我們 OPEN 這個檔，建立起聯繫的關係，接著就指出我們所要建立的關係是 WRITE ；這時候我們所要儲存的項目就是在這個檔中記錄的數目，目前就是零；敍述 1140 做的就是這個工作。最後的工作則是關閉與 DOS 間的聯繫，而這就由行號 1150 的敍述完成了，現在我們已經建立了這個檔。

我們現在已經可以將名字輸入這個檔內了，這是由 3000 到 3999 的程式行所執行的。3000 到 3270的程式段就是使您能夠輸入名字，而在您輸入完畢之後，位於 3270 到 3360 的程式行會將這些名字寫入這個檔內。

第一個我們們要做的事就是 DELETE 原先的檔，這麼做的原因是因爲我們使用同樣的程式段輸入、改變及消除名字；由於在每一個記錄之後都有一個游標回頭(carriage return)字，如果在一個記錄內的資料有了改變而我們不是由頭開始做寫入的工作，那麼 DOS 3.3 將新的資料寫入檔內並且留下這個游標回頭字；因此，當您下一次要讀取這個名字的時候可能就會不正確了。我們爲了避免這種情況，因此先將原先的檔消除掉。

接著這個檔再被打開，現在我們就要把資料寫入檔內了。首先我們將檔內記錄的數目, XI ，寫入檔內；行號 3320 的敍述做的就是這個工作；接著我們就把這個班級內所有學生的名字寫入這個檔內了，這一段的工作則是由行號 3330 到 3350 的程式行完成的；注意行號 3340 內的 PRINT 敍述，它將班上學生的名字， CL$(I) ，寫入檔內，這個敍述後面跟著一個 RETURN 就像在螢光幕顯示時同樣的情況，這個 RETURN 就做爲記錄間的分隔，請參看圖 8 — 9。

假設您在今天輸入完畢一些名字，而在明天又想輸入更多的名字到這個檔內，當您執行這個 CLASS LIST 的程式時，第一件您被詢問的就是您想要使用的檔的名字，在目前的情況下就是 4TH GRADE 而我們就再一次的打開了這個檔，但是這一次我們所要做的是讀取這個檔內的資料，首先要做的就是 INPUT R，輸入原先存在這個檔內的名字的數目，然後我們就可以使用這個訊息將每一個學生的名字都讀入記憶體內了，然後再把這個檔關閉。這些工作都是在行號 835 到 870 的敍述內完成的。

請注意我們分別在行號 830 及 875 的程式行打開及關閉了DOS 處理錯誤的程式段，而這個程式段就位於行號 8000 到 8430 內。

8.2.2.2 隨機處理檔

隨機處理檔與順序處理檔是不相同的，您可以任意處理一個已經被寫入磁片內的記錄；同時，這種型式的檔儲存在磁片上的格式也不相同，這種檔是由記錄所組成的，而每個記錄內包含著許多的區(fields)，區內存的就是數字或文字。圖 8 — 10。就是這種檔的組成情形。

```
                 ********
                 *      *
                 *  檔  *
                 *      *
                 ********
                     !
        +------------+------....-----+
        !            !               !
  **********     **********      **********
  *        *     *        *      *        *
  *  記錄  *     *  記錄  * .... *  記錄  *
  *        *     *        *      *        *
  **********     **********      **********
                     !
                     !
        +------------+-----....-----+
        !            !              !
  *********      *********      *********
  *       *      *       *      *       *
  *  區   *      *  區   * .... *  區   *
  *       *      *       *      *       *
  *********      *********      *********
                     !
                     !
        +------------+-----....-----+
        !            !              !
  *********      *********      *********
  *       *      *       *      *       *
  *  字   *      *  字   * .... *  字   *
  *       *      *       *      *       *
  *********      *********      *********
```

圖 8 — 10 層次型的檔的結構

這種型式的檔還有一些比較特殊的地方，那就是每一個記錄都是相同大小、長度，而且在正常的情況下，存在記錄內的資料都是以同樣的次序及組成方式安排的，而每一個記錄內的每一個區部可以隨意的處理。

隨機處理檔是非常有用的，但是它比較浪費磁片的儲存位置；因此，如果您想要使用它的好處，那麼您無法兼顧節省儲存位置的問題。

隨機處理檔的結構需要您指出一些元素；第一個就是記錄的長度，由於檔內所有的記錄都是相同的長度，因此 DOS 3.3 在您寫入一個新的記錄之後會自動地替您準備好這個長度的儲存位置；其次，您必須指出您所要寫入的是那一個記錄。當您在打開檔與 DOS 之間的聯繫的同時，您必須指出檔的名字及記錄的長度，同時您也可以指出擴充接點，磁碟機及磁片號碼等元素，但是它們都是可以被省略的。而當您要處理這個檔的時候，不論是讀還是寫，您也必須指出檔的名字及記錄的號碼，至於擴充接點，磁碟機及磁片號碼等元素則是可有可無的。

圖 8 — 11 就是一個我們稱它爲 GRADE BOOK 的程式，它就是使用隨機處理檔處理資料的說明。這個程式只是一個簡單的分數表，它可以處理 4 個學期內的成績，其中的三個就是期中的考試成績，第四個則是期末測驗的成績，至於第 5 個成績則是學期的平均分數，我們在這個程式中並不考慮加重期末測驗成績的比例，或是增加其他的分數。每一個分數佔用了五個數元組的儲存位置，至於名字則允許您輸入 25 個字。如果您想修改這個程式您也可以依您的意思做，這可以做爲您的一個練習。每增加一個分數就必須增加五個數元組的記錄長度，同樣地，在求平均分數的副程式部份您也必須加以修改。

```
1  REM *** GRADE BOOK ***
2  REM *
3  REM * WRITTEN BY:JL CAMPBELL
4  REM *        DATED:07/22/81
5  REM *
6  REM *    LANGUAGE:APPLESOFT
7  REM *    COMPUTER:APPLE II
8  REM *
9  REM ****************
10  TEXT : CLEAR : POKE 216,0:A =  FRE (0)
20  DIM CL$(50),G(50,5)
```

```
45 D$ =  CHR$ (4)
50  GOTO 800
51  REM **************
52  REM * GEN'L SUBR *
53  REM **************
55  INPUT "WHICH ONE? ==> ";N: RETURN
60  VTAB 22: CALL  - 958: HTAB 13: INVERSE : PRINT
"ILLEGAL ENTRY!"; CHR$ (7): NORMAL
64  FOR L = 1 TO 1000: NEXT : RETURN
70  VTAB 4: CALL  - 958:IX = R
75  PRINT "THERE ARE ";R;" STUDENTS IN ";F$: RETURN
80  INPUT "TYPE 'RETURN' KEY TO CONTINUE....";A$:
RETURN
99  REM * NAME Y/N SUBR
100 SW = 0
105  VTAB 20: INPUT "ANOTHER NAME?(Y/N) = ";A$
110  IF  LEFT$ (A$,1) = "Y" THEN SW = 1: RETURN
115  IF  LEFT$ (A$,1) = "N" THEN SW = 0: RETURN
120  GOSUB 60: GOTO 100
124  REM * STUDENT SUBR
125  GOSUB 70
130  VTAB 6: CALL  - 958
135  INPUT "WHICH STUDENT #? = ";RN
140  VTAB 8: PRINT "STUDENT #";RN;" = ";CL$(RN)
145  RETURN
149  REM * PRINT GRADES SUBR
150  VTAB 10: CALL  - 958
155  FOR I = 1 TO 5
160  PRINT "GRADE #";I;" = ";G(RN,I)
165  NEXT I
170  VTAB 16: PRINT "0- EXIT"
175  VTAB 18: CALL  - 958: GOSUB 55
180  RETURN
199  REM * GRADE AVG SUBR
200 T = 0:C = 0
205  FOR I = 1 TO 4
210  IF G(RN,I) = 0 THEN 220
215 T = T + G(RN,I):C = C + 1
220  NEXT I
225 G(RN,5) =  INT (T / C * 100 + 0.5) / 100
230  RETURN
599  REM * PRINTER SETUP SUBR
600  PRINT "DO THE FOLLOWING:"
605  PRINT  TAB( 5)"1- LOAD PRINTER PAPER"
610  PRINT  TAB( 5)"2- TURN ON PRINTER"
615  PRINT  TAB( 5)"3- SET TOP-OF-FORM"
620  PRINT  TAB( 5)"4- PRINTER ON-LINE"
```

```
625  PRINT : INPUT "TYPE 'RETURN' KEY TO CONTINUE...
";A$
630  VTAB 12: CALL  - 958
635  INPUT "ENTER PRINTER SLOT(1-7) = ";SL:SL =  INT
(SL)
640  IF SL > 0 AND SL < 8 THEN 650
645  GOSUB 60: GOTO 630
650  VTAB 14: CALL  - 958
655  PRINT "PRINTER TYPE:"
660  PRINT  TAB( 5)"1- SERIAL"
665  PRINT  TAB( 5)"2- PARALLEL"
670  PRINT : GOSUB 55:SP =  INT (N)
675  IF SP = 1 OR SP = 2 THEN 685
680  GOSUB 60: GOTO 650
685  PR# SL
690  IF SP = 2 THEN  PRINT  CHR$ (9);"80N"
695  RETURN
699  REM * WRITE SUBR
700  PRINT D$;"OPEN ";F$;",L50,V,S6,D1"
705  PRINT D$;"WRITE ";F$;", R0"
710  PRINT IX
715  IF IX = 0 THEN 750
720  FOR I = 1 TO IX
725  PRINT D$;"WRITE ";F$;", R";I
730  PRINT CL$(I)
735  FOR J = 1 TO 5
740  PRINT G(I,J)
745  NEXT J,I
750  PRINT D$;"CLOSE";F$
755 R = IX
780  RETURN
800  HOME : PRINT : PRINT  TAB( 12)"*** FILE DATA ***":
PRINT
802  PRINT "TYPE 'RETURN' KEY IF FILE NOT CREATED!"
805  VTAB 6: CALL  - 958
810  INPUT "ENTER FILE NAME = ";F$
815 L =  LEN (F$): IF L < 2 THEN 900
820  IF L > 20 THEN F$ =  LEFT$ (F$,20)
825 F$ = "GRADE." + F$
830  ONERR  GOTO 8000
835  PRINT D$;"OPEN ";F$;",L50,V,S6,D1"
840  PRINT D$;"READ ";F$;", R0"
845  INPUT R
850  IF R < 1 THEN 890
855  FOR I = 1 TO R
860  PRINT D$;"READ ";F$;", R";I
```

```
865  INPUT CL$(I)
870  FOR J = 1 TO 5
875  INPUT G(I,J)
880  NEXT J,I
890  PRINT D$;"CLOSE ";F$
895  POKE 216,0
899  REM * MAIN DRIVER
900  HOME : PRINT : PRINT  TAB( 9)"*** GRADE BOOK PROG
***": PRINT
905  PRINT "MAIN MENU:": PRINT "---- ----"
910  PRINT "1- CREATE GRADE FILE"
912  PRINT "2- DELETE GRADE FILE": PRINT
914  PRINT "3- ENTERS NAMES";: HTAB 20: PRINT " 8-
ENTERS GRADES"
916  PRINT "4- CHANGE NAMES";: HTAB 20: PRINT " 9-
CHANGE GRADES"
918  PRINT "5- DELETE NAMES";: HTAB 20: PRINT "10-
DELETE GRADES": PRINT
920  PRINT "6- REVIEW NAMES";: HTAB 20: PRINT "11-
REVIEW GRADES"
922  PRINT "7- PRINTS NAMES";: HTAB 20: PRINT "12-
PRINTS GRADES": PRINT
928  PRINT "0- EXIT PROGRAM"
930  VTAB (18): CALL  - 958: GOSUB 55:N =  INT (N)
935  IF N = 0 THEN 990
940  IF N > 0 AND N < 13 THEN 950
945  GOSUB 60: GOTO 930
950  ON N GOSUB 1000,2000,3000,4000,5000,
      6000,7000,3500,4500,5500,6500,7500
960  GOTO 900
990  HOME : CALL  - 958: VTAB 10: PRINT "GOODBYE FOR
NOW...."
995  END
1000  HOME : PRINT : PRINT  TAB( 9)"*** CREATE FILE
***": PRINT
1010  PRINT "NAME LESS THAN 20 CHARS."
1020  VTAB 6: CALL  - 958
1030  INPUT "ENTER FILE NAME = ";F$
1040 L =  LEN (F$): IF L < 2 THEN 1999
1050  IF L > 20 THEN F$ =  LEFT$ (F$,20)
1060 F$ = "GRADE." + F$:N = 0
1100  PRINT D$;"OPEN ";F$;",L50,V,S6,D1"
1110  PRINT D$;"DELETE ";F$
1120  PRINT D$;"OPEN ";F$;",L50,V,S6,D1"
1130  PRINT D$;"WRITE ";F$;", R0"
```

```
1140  PRINT N
1150  PRINT D$;"CLOSE ";F$
1999  RETURN
2000  HOME : PRINT : PRINT  TAB( 9)"*** DELETE FILE
***": PRINT
2100  PRINT D$;"DELETE ";F$
2999  RETURN
3000  HOME : PRINT : PRINT  TAB( 10)"*** ENTERS NAMES
***": PRINT
3010  GOSUB 70
3020 IX = R
3030  VTAB 6: CALL  - 958: VTAB 10
3040 IX = IX + 1:R = IX
3050  PRINT "ENTER NAME #";IX;: INPUT " = ";CL$(IX)
3060  GOSUB 100
3070  IF SW = 1 THEN 3000
3080  GOSUB 700
3090  RETURN
3500  HOME : PRINT : PRINT  TAB( 10)"*** ENTERS GRADE
***": PRINT
3510  GOSUB 125
3520  GOSUB 150
3530  IF N = 0 THEN 3590
3540  IF N > 0 AND N < 6 THEN 3560
3550  GOSUB 60: GOTO 3600
3560  VTAB 20: PRINT "GRADE #";N;: INPUT " = ";G(RN,N)
3570  GOSUB 200
3580  GOTO 3520
3590  GOSUB 100
3600  IF SW = 1 THEN 3500
3610  GOSUB 700
3620  RETURN
4000  HOME : PRINT : PRINT  TAB( 10)"*** CHANGE NAMES
***": PRINT
4010  GOSUB 125
4020  VTAB 10: INPUT "CHANGE TO WHAT? = ";CL$(RN)
4030  GOSUB 100
4040  IF SW = 1 THEN 4000
4050  GOSUB 700
4060  RETURN
4500  HOME : PRINT : PRINT  TAB( 10)"*** CHANGE GRADE
***": PRINT
4510  GOSUB 125
4520  GOSUB 150
```

```
4540  IF N = 0 THEN 4600
4550  IF N > 0 AND N < 6 THEN 4570
4560  GOSUB 60: GOTO 4520
4570  VTAB 20: PRINT "GRADE #";N;: INPUT " = ";G(RN,N)
4580  GOSUB 200
4590  GOTO 4520
4600  GOSUB 100
4610  IF SW = 1 THEN 4500
4620  GOSUB 700
4630  RETURN
5000  HOME : PRINT : PRINT  TAB( 10)"*** DELETE NAMES
***": PRINT
5010  GOSUB 125
5020  VTAB 10: PRINT "DELETE STUDENT ";CL$(RN);: INPUT
"?(Y/N) ";A$
5030  IF  LEFT$ (A$,1) = "Y" THEN 5060
5040  IF  LEFT$ (A$,1) = "N" THEN 5000
5050  GOSUB 60: GOTO 5020
5060  FOR I = RN + 1 TO R
5070 CL$(I - 1) = CL$(I)
5080  FOR J = 1 TO 5:G(I - 1,J) = G(I,J): NEXT J
5090  NEXT I
5100 CL$(R) = "    "
5110  FOR J = 1 TO 5:G(R,J) = 0: NEXT J
5120 R = R - 1:IX = R
5130  IF R = 0 THEN 5180
5140  GOSUB 100
5150  IF SW = 1 THEN 5000
5160  GOSUB 700
5170  RETURN
5180  VTAB 4: CALL  - 958
5190  PRINT "THERE ARE ZERO STUDENTS STORED."
5200  PRINT "RECREATING THE FILE TO REFLECT THIS!"
5210  PRINT : PRINT "PLEASE WAIT A MOMENT."
5220  GOSUB 700
5230  RETURN
5500  HOME : PRINT : PRINT  TAB( 10)"*** DELETE GRADE
***": PRINT
5510  GOSUB 125
5520  GOSUB 150
5530  IF N = 0 THEN 5590
5540  IF N > 0 AND N < 6 THEN 5560
5550  GOSUB 60: GOTO 175
5560 G(RN,N) = 0
5570  GOSUB 200
```

```
5580  GOTO 5520
5590  GOSUB 100
5600  IF SW = 1 THEN 5500
5610  GOSUB 700
5620  RETURN
6000  HOME : PRINT : PRINT  TAB( 10)"*** REVIEW NAMES
***": PRINT
6010  FOR I = 1 TO R STEP 10
6020  VTAB 4: CALL  - 958
6030  FOR J = I TO I + 9
6040  IF  LEFT$ (CL$(J),1) = "" OR  LEFT$ (CL$(J),1) =
" " THEN 6060
6050  PRINT J;"- ";: HTAB 5: PRINT CL$(J)
6060  NEXT J
6070  VTAB 18: GOSUB 80
6080  NEXT I
6090  RETURN
6500  HOME : PRINT : PRINT  TAB( 10)"*** REVIEW GRADE
***": PRINT
6510  FOR I = 1 TO R STEP 10
6520  FOR J = I TO I + 9
6530  VTAB 4: CALL  - 958
6540  IF  LEFT$ (CL$(J),1) = "" OR  LEFT$ (CL$(J),1) =
" " THEN 6610
6550  PRINT J;"- ";: HTAB 5: PRINT CL$(J)
6560  VTAB 6
6570  FOR K = 1 TO 5
6580  PRINT "GRADE #";K;" = ";G(J,K)
6590  NEXT K
6595  VTAB 10: HTAB 18: PRINT "= AVERAGE"
6600  VTAB 12: GOSUB 80
6610  NEXT J
6620  VTAB 18: GOSUB 80
6630  NEXT I
6640  RETURN
7000  HOME : PRINT : PRINT  TAB( 10)"*** PRINTS NAMES
***": PRINT
7010  GOSUB 600
7020  PRINT  TAB( 28)"*** CLASS LIST REPORT ***": PRINT
" "
7030  PRINT  TAB( 10)"CLASS LIST FOR ";F$;" FILE."
7040  PRINT " "
7050  PRINT  TAB( 11)"#";: POKE 36,20: PRINT "NAME"
7060  PRINT  TAB( 10)"---";: POKE 36,20: PRINT "----"
7070  PRINT " "
```

```
7080  FOR I = 1 TO R
7090  PRINT  TAB( 10)I;: POKE 36,13: PRINT "-";
7100  POKE 36,20: PRINT CL$(I)
7110  NEXT I
7120  PRINT  CHR$ (12)
7130  IF SP = 2 THEN  PRINT  CHR$ (9);"40N"
7140  PR# 0
7150  RETURN
7500  HOME : PRINT : PRINT  TAB( 10)"*** PRINT GRADES
***": PRINT
7510  GOSUB 600
7520  PRINT  TAB( 28)"*** CLASS GRADE REPORT ***":
PRINT
7530  PRINT  TAB( 10)"GRADE LIST FOR ";F$;" FILE."
7540  PRINT " "
7550  PRINT  TAB( 5)"#";: POKE 36,10: PRINT "NAME";
7555  POKE 36,40
7560  FOR I = 1 TO 4: PRINT I;"        ";
7565  NEXT I: PRINT "AVG."
7570  PRINT  TAB( 4)"---";: POKE 36,10: PRINT "----";
7575  POKE 36,40
7580  FOR I = 1 TO 4: PRINT "-       ";
7585  NEXT I
7590  PRINT "----": PRINT " "
7600  FOR I = 1 TO R: POKE 36,4
7610  PRINT I;: POKE 36,7: PRINT "-  ";
7620  PRINT CL$(I);
7630  FOR J = 1 TO 5
7640  POKE 36,8 * (J - 1) + 40
7650  PRINT G(I,J);
7660  NEXT J: PRINT : POKE 36,0
7670  NEXT I
7680  PRINT  CHR$ (12)
7970  IF SP = 2 THEN  PRINT  CHR$ (9);"40N"
7980  PR# 0
7990  RETURN
7999  REM * ERROR ROUTINES
8000 E =  PEEK (222): VTAB 20: CALL  - 958
8010  PRINT "ERROR = <";E;"> ";
8020  POKE 222,0
8030  ON E GOTO 8100,8150,8150,8200,8250,
       8300,8100,8100,8350,8400
8100  PRINT "CONSULT MANUAL"
```

```
8110  GOSUB 64: GOTO 800
8150  PRINT "RANGE ERROR"
8160  GOSUB 64: GOTO 800
8200  PRINT "WRITE PROTECTED"
8210  VTAB 22: CALL  - 958: PRINT "REMOVE PROTECTION.
TYPE 'RETURN' KEY."
8220  GET K$: IF  ASC (K$) = 13 THEN  PRINT : GOTO 800
8230  GOSUB 60: GOTO 8210
8250  PRINT "FILE NOT CREATED"
8270  GOSUB 64: GOTO 900
8300  PRINT "FILE NOT FOUND"
8310  GOSUB 64: GOTO 800
8350  PRINT "DISK FULL"
8360  VTAB 22: CALL  - 958: PRINT "CHANGE DISKETTES.
TYPE 'RETURN' KEY."
8370  GET K$: IF  ASC (K$) = 13 THEN  PRINT : GOTO 800
8380  GOSUB 60: GOTO 8360
8400  PRINT "FILE LOCKED"
8410  VTAB 22: CALL  - 958: PRINT "UNLOCK GRADE FILE."
8420  FOR I = 1 TO 10: PRINT : NEXT I
8430  END
```

圖8-11 GRADE BOOK程式

```
                    *** CLASS LIST REPORT ***

    CLASS LIST FOR GRADE.CS-101

     #          NAME
    ---         ----

    1  -        JOHN CAMPBELL
    2  -        LANCE ZIMMERMAN
    3  -        MITCHEL WYATT
    4  -        DEBBIE WALTERS
    5  -        RICHARD CRANSTON
    6  -        TOM TURNER II
    7  -        BARBARA FEATHER
    8  -        GEORGE R. SMITH
    9  -        CYNTHIA BOGGS
    10 -        HAROLD P. WINDSOR
    11 -        KATHY VIVIAN
```

圖8-12 CLASS LIST報表

```
                       *** CLASS GRADE REPORT ***

      GRADE LIST FOR GRADE.CS-101 FILE.

#       NAME                   1      2      3      4      AVG.
---     ----                   -      -      -      -      ----

1   -   JOHN CAMPBELL          95.75  96     0      0      95.88
2   -   LANCE ZIMMERMAN        85.55  90     0      0      87.78
3   -   MITCHEL WYATT          87.89  84.5   0      0      86.2
4   -   DEBBIE WALTERS         87.34  91.25  0      0      89.3
5   -   RICHARD CRANSTON       72     88.75  0      0      80.38
6   -   TOM TURNER II          68     88     0      0      78
7   -   BARBARA FEATHER        99.75  0      0      0      99.75
8   -   GEORGE R. SMITH        69.95  75     0      0      72.48
9   -   CYNTHIA BOGGS          87.56  92.5   0      0      90.03
10  -   HAROLD P. WINDSOR      70     88.75  0      0      79.38
11  -   KATHY VIVIAN           95     100    0      0      97.5
```

圖 8-13　CLASS GRADE 報表

在圖 8－11 中的是一個非常直接的程式，程式的邏輯就是圖 4－8 中的架構；程式中只有三個區域較爲有趣，它們是行號 700 到 780 的寫的副程式，行號 835 到 895 的取資料的程式段，以及行號 1100－1150 的建立新檔的程式段。至於程式的其他部份我們相信您目前已經可以瞭解它們了。因此我們也就不再詳述。這個檔的名字是 CS-101 做爲我們說明的例子。

我們首先由行號 1100 到 1150 的程式段做說明。

行號 1100－打開與 DOS 3.3 間的聯繫。
－指出檔的名字。
－將這個檔在 VTOC 中建立起來。
－設定記錄的長度爲 50 個數元組。
－不管磁片號碼。
－設定擴充接點的數目爲 6。
－設定磁碟機號碼爲 1。

行號 1110－消除剛建立的檔；但是將它的名字留在 VTOC 內。

行號 1120－重新打開與 DOS 3.3 的聯繫。

－如行號 1100 般的建立這個檔。

行號 1130－告訴 DOS 寫是聯繫的型式。

－指出檔的名字。

－要把資料寫入第零個記錄。

行號 1140－將記錄的數目存入檔內。

－存的位置就是記錄零。

行號 1150－將檔與 DOS 3.3 間的聯繫關閉。

行號 1999－返回主程式。

行號 700－780 的程式段將整個的檔寫入磁片內，我們將新的記錄數目存在第零個記錄內。然後將每個學生的名字，CL$，及這個學生的成績，G，存入每一個記錄內。由於大多數的班級都不太大，所以我們每次都重新將所有的記錄寫入磁片內，當然我們也可以只把修改過或是新增加的資料寫入檔內。

行號 700－打開與 DOS 3.3 的聯繫。

－指出檔的名字。

－在 VTOC 內建立這個檔。

－設定記錄的長度是 50 個數元組。

－不管磁片號碼。

－設定擴充接點為 6。

－設定磁碟機號碼為 1。

行號 705－設定聯繫的型式為寫。

－再指出檔的名字。

－指定第零個記錄。

行號 710－寫入第零個記錄的資料。

－存入學生的總數。

行號 715－測試檔內第零個記錄的資料。

　　　－如果是零，則關閉這個檔。

行號 720－ FOR...NEXT 的廻圈用來寫入每個記錄。

行號 725－設定聯繫的型式爲寫。

　　　－指出檔的名字。

　　　－使用脚註(index)I指到每個記錄。

行號 730－將名字存入第 I 個記錄內。

行號 735－ FOR...NEXT 的廻圈將每一個成績寫入記錄I 內。

行號 740－將每一個成績存入記錄 I 內。

行號 745－對於兩個廻圈範圍的設定。

行號 750－將檔與 DOS 3.3 間的聯繫關閉。

行號 780－返回主程式。

位於行號 830到895的程式段將已經設定的檔讀入記憶體內；注意我們在這裏打開了處理錯誤的副程式，這個副程式的目的是要找到並不存在的檔或是做資料結束時的反應等等工作。檔的名字由鍵盤在行號 810的地方輸入，接著我們就就把 "GRADE." 加在名字的前面。

行號 830－打開處理錯誤的副程式。

行號 835－打開與 DOS 3.3 之間的聯繫。

　　　－指出檔的名字。

　　　－在 VTOC 內建立這個檔。

　　　－設定記錄的長度爲 50 個數元組。

　　　－不管磁碟片號碼。

　　　－設定擴充接點的數目爲 6 。

　　　－設定磁碟機號碼爲 1 。

行號 840－設定聯繫的型式爲 READ 。

　　　－指出檔的名字。

　　　－指出第零個記錄是要被讀取的。

行號 845 — 取得檔內記錄的數用。

行號 850 — 如果檔內沒有記錄則關閉這個檔。

行第 855 — FOR...NEXT 廻圈用來讀取所有的記錄。

行號 860 — 設定聯繫的型式為 READ 。

— 指出檔的名字。

— 指出檔中的第 I 個記錄為處理對像。

行號 865 — 取得第 I 個記錄內的學生名字。

行號 870 — FOR...NEXT 的廻圈用來讀取記錄 I 內的每一個分數。

行號 875 — 取得記錄 I 內的每一個成績。

行號 880 — 對於兩個廻圈範圍的設定。

行號 890 — 關閉與 DOS 3.3 之間的聯繫。

行號 895 — 關閉處理錯誤的副程式。

我們行以在處理錯誤及檢查範圍的工作上做得更好一些，但是這個程式只是做為一個示範，其餘的工作您可以自己加入。

圖 8 — 12 及 8 — 13 就是這個 GRADE BOOK 程式所產生出來的兩份報告，這些報告就是存在檔內的資料被顯示或計算後產生的結果。

目前我們已經告訴您基本 DOS 3.3 的命令了，同時您也知道如何使用順序處理檔及隨機處理檔處理資料了。您可以參看附錄D更進一步瞭解這兩種型式的檔，每一種型式都各有它的優缺點，至於如何應用就必須由您在瞭解了問題的性質之後做決定了。

到現在為止我們所介紹的處理檔的命令如下：

OPEN — 建立與 DOS 3.3 的聯繫。

READ — 聯繫的型式。

由磁片上取出資料。

WRITE — 聯繫的型式。

將資料存到磁片上。

INPUT — 將資料存入執行的程式中。

PRINT　—將資料存入磁碟檔內。

CLOSE　—結束與 DOS 3.3 間的聯繫。

8.2.2.3 二進位檔

有時候也許您要設計機器語言的程式，那麼如果 DOS 3.3 能夠讓您將您的程式以一個二進位檔的型式存在磁片上那不是很好嗎？幸運的是，DOS 3.3 就能夠支持您這個構想。下面的幾個命令就是幫助您做這個工作。

BSAVE　檔的名字，A 位置，L 長度。

BLOAD　檔的名字（ ，A位置 ）

BRUN　檔的名字（ ，A 位置 ）

注意：擴充接點，磁碟機及磁片號碼都是可有可無。

顯然地，在這些命令前面的字就代表著二進位檔的意思。當您BSAVE 一個檔，那麼您就是把記憶體內這個檔用十六進位數值的表示方法存入磁片內；因此，您必須指出這個程式在記憶體內的起始位置以及這個程式的長度；位置（address）可以用十進位或是十六進位數字表示，而長度元素也是同樣的情況。請參看附錄D詳細地瞭解這些命令的格式。在第四章的 4.1 節中我們曾經顯示過了使用十六進位及十進位數字的格式。在 BSAVE 命令中位置及長度元素都不可以省略。下面就是一些 BSAVE 的例子：

```
]BSAVE HI-RES.PIC,A$2000,L$2000

]BSAVE HI-RES.PIC,A16384,L8192

]BSAVE HI-RES.PIC,A$2000,L8192

]BSAVE HI-RES.PIC,A16384,L$2000
```

圖 8-14　BSAVE的例子

如果您想由磁片將一個二進位檔存到記憶體內，那麼您就可以使用BLOAD命令做這個工作；下面是一些使用這個命令必備條件以及需要注意的地方。首先， BLOAD 命令需要一個位置元素，這個位置就是存入記憶體內的二進位檔開始存放的地方；其次，您 BLOAD 的這個檔未必能在不同於 BSAVE 時所指明的位置上執行，下面就是幾個相等的命令：

```
]BLOAD HI-RES.PIC,A$2000
]BLOAD HI-RES.PIC,A16384
```

圖 8 -15 BLOAD的例子

最後一個二進位檔的命令就是 BRUN ；這個命令首先會執行一個BLOAD命令，然後再執行被存入記憶體內的程式。如果您指出一個位置，那麼這個程式就會被存入這個位置開始的記憶體內，然後再被執行。下面就是一個例子：

```
]BRUN SPACE GAME,S6,D1
```

圖 8 -16 BRUN例子

B 字代表的就是二進位檔；目前我們已經知道了四種不同型式的檔了。它們分別是：

* ＊－A’Applesoft][程式。
* ＊－I’整數 BASIC 程式
* ＊－B’文字資料檔
* ＊－T’二進位檔

8.2.2.4 EXEC檔

DOS 3.3 中的 EXEC 命令具有 RUN 命令的特性，但是它所執行

的是一個文字資料檔（text　file)而不是一個程式。文字資料檔內可以存有命令，程式行，　BASIC　敍述等等。

因此我們可以建立一個順序處理的文字資料檔，檔內存的就是合法的 Applesoft　][　，命令。這個命令的格式如下：

EXEC 檔的名字（’R位址）

現在我們假設您想建立一個 EXEC　的檔，而這個檔會自動地執行一個程式並將它列印在螢光幕上。處理這個問題的方式非常直接，對於我們的範例程式而言我們將使用前面 8.2.1.2節中圖 8－4的　MATH　DEMO（後改爲　DUMMY　）程式做爲示範。下面是我們要做的工作。

1－設計一個處理順序處理檔的程式，具有；

＊——個新的名字，　DEMO EXEC

＊－將它與　DUMMY　程式存在同一片磁片內。

＊－程式建立一個名叫 EXAMPLE　的文字資料檔，

＊－執行 DEMO EXEC　。

2－ EXEC EXAMPLE

```
1 REM ***** DEMO EXEC *****
2 REM *
3 REM * WRITTEN BY:JL CAMPBELL
4 REM *       DATED:08/08/81
5 REM *
6 REM * LANGUAGE:APPLESOFT II
7 REM * COMPUTER:APPLE II PLUS
8 REM *
9 REM **********************
10 TEXT : CLEAR : POKE 216,0 : A = FRE(0)
20 HOME : PRINT : PRINT TAB( 11)"*** DEMO EXEC ***"
: PRINT
30 D$ = CHR$(4) : REM CTRL-D
40 PRINT D$;"OPEN EXAMPLE,V,S6,D1"
50 PRINT D$;"DELETE EXAMPLE"
```

```
60 PRINT D$;"OPEN EXAMPLE"
70 PRINT D$;"WRITE EXAMPLE"
80 PRINT "RUN DUMMY"
90 PRINT "LIST 1,110"
100 PRINT D$;"CLOSE EXAMPLE"
110 END
```

圖 8-17 DEMO EXEC程式

```
]SAVE DEMO EXEC,D1

]RUN
```

我們現在就可以執行這個剛剛建立的文字資料檔 EXAMPLE 了。

```
]EXEC EXAMPLE,D1
```

下面有一些事情您必須注意。第一，存在 EXAMPLE 這個文字資料檔內的記錄並不具有行號；第二，每一個記錄是一個合法的命令，第三，必須先執行建立文字資料檔的程式以便將這個文字資料檔存到磁片上去；最後，您必須 EXEC 這個文字資料檔而得到最後的結果。

也有一些情況您必須注意的，那就是在 EXEC 命令執行的時候，CTRL-C 這個打斷程式執行的命令並不發生作用，一直要到 DUMMY 程式執行時， CTRL-C 才能再度的發生效果。

下面是一些使用 EXEC 執行文字資料檔的好處：例如；

* 一將整數 BASIC 程式轉換成 Applesoft][程式。
* 一修正程式。
* 一將程式段按入程式中。
* 一重新設定程式的行號。
* 一將程式行移到文字資料檔內，再移到 APPLEWRITER 的檔內。
* 一建立一個自動的，轉鍵（turnkey）的系統而不需要您去觀察。

8.3 總　論

這一章告訴了您 Apple][電腦上的作業系統。我們也使用另外一種觀點介紹了電腦邏輯上的結構。

接著我們對於磁帶的作業系統做了大致地討論；由於使用磁帶儲存資料或程式的速度實在太慢了，因此我們所討論的也就足夠了。

至於隨後的幾節我們都專注於 DOS 3.3 的討論；在 8.2.1節中我們介介紹了一些使用磁碟系統的命令。它們包括了：

]INIT	檔的名字	例如	] INIT HELLO
]SAVE	檔的名字	例如	] SAVE MATH DEMO
]LOAD	檔的名字	例如	] LOAD DUMMY
]CATALOG		例如	] CATALOG,D2
]RENAME	檔 1 ，檔 2	例如	]RENAME MATH DEMO,DUMMY
]DELETE	檔的名字	例如	]DELETE GENERAL LEDGER
]LOCK	檔的名字	例如	] LOCK DUMMY
]UNLOCK	檔的名字	例如	] UNLOCK HELLO
>FP		注意：轉移到 APPLESOFT][	
]INT		注意：轉移到整數 BASIC	

我們接著討論的是磁片的格式，儲存的組織方式， VTOC ，以及磁軌/磁區表，最後則是 DOS 3.3 及其他型式 DOS 的特性。

檔的結構我們與實際的處理狀況同時討論；對於 DOS 3.3 所支持的兩種型式的結構，我們都做了相當詳細的說明，範例程式中也都使用了建檔，取資料等等的 DOS 3.3 的命令。這些命令分別是：

OPEN	CLOSE
READ	WRITE

INPUT PRINT

每一個命令都因爲檔型式的不同而有稍許的差別，這些差別的情形您在研究過了範例程式之後相信已經能夠瞭解了，我們非常希望您能夠確實地瞭解處理不同型式檔的命令。

接下來的一節討論的是儲存，取用及執行二進位檔的命令，它們分別是:

BSAVE BLOAD BRUN

最後討論的項目就是 EXEC 命令；這個命令允許您建立一個能夠自動處理的文字資料檔。

在這一章中，我們討論了一些您將會經常使用的命令，我們並沒有討論像是 POSITION, BYTE 等等的問題，那是因爲我們覺得您已經能夠自己參看手册就可以瞭解它們了；而您也能夠使用這些 DOS 3.3 的命令處理您的問題了。

第 9 章

圖 形

9.0 大 綱

這一章討論的是 Apple][顯示圖形的能力。 Apple][顯示的圖形有兩種——低解析度的及高解析度（ low resolution and high resolution ）的圖形。

這種顯示圖形的能力就是在市場上大多數遊戲程式的基礎，較精細的圖形使用的就是高解析度圖形的能力，因爲高解析度圖形表現出來的畫面較低解析圖形要精細得多了。我們也在許多的應用程式中使用這種能力繪出各種像是函數圖形、庫存及市場的升降表等等的圖表。

我們將告訴您低解析度圖形及高解析度圖形的能力並對這些能力分別做詳細的說明；在第十一章中我們將再度的顯示幾個低解析度圖形的例子。

這一章中同時也討論了將圖形印在印表機上的問題，我們將對不同型式的印表機做詳細地說明並且告訴您如何在印表機上印出圖形。

9.1 低解析度圖形

低解析度圖形就如它的名字所顯示的，它在螢光幕上繪出的“點”是非常大的。 Apple 在正常情況下能夠顯示 24 列，每列 40 個字的文字資料，每一個字佔用了 7 個水平及 8 個垂直顯示點的位置，這使得螢幕能夠顯示的點總共有 280 × 192 個。

至於低解析度的圖形則可以在螢幕上顯示出 40 乘上 48 個彩色“點”，而正常情況下的顯示幕則是 40 乘 40，剩下的螢幕則用來顯示 4 列的文字資料。每一個低解析度的“點”佔用了 7 個水平顯示點及 4 個垂直顯示點，由於每一個“點”佔用了 28 個顯示點，因此這種圖形的解析度實在是非常低的；因此，我們稱它爲低解析度圖形。

我們進入低解析度圖形的命令是 GR ，一般都是在間接式情況下輸入的，這個命令建立了一個 40 列乘上 40 行的圖形顯示幕，並且在螢幕的底端留下四列的文字資料行。

計算顯示幕位置的方法是由左至右分別用 0 到 39 的數字表示，至於垂直方向則由上到下也是用 0 到 39 的數字代表，在螢幕上繪出一個點可以使用的方法很多，也可以將許多的點連接在一起而繪出綫來。請參看圖 9－1。

```
10 GR : COLOR = 6
30 HLIN 0,39 AT 0 : VLIN 0,39 AT 0
40 VLIN 0,39 AT 39 : HLIN 0,39 AT 39
50 END
```

圖9－1　低解析度圖形的例子

圖 9－1 中的程式將延著圖形顯示幕的四週繪出一個藍色的正方型；需要注意的是，您必須在繪圖形之前先設定顏色值。

您在繪圖的時候可以使用 16 種不同的顏色在螢光幕上繪出圖形；當然，您必須具備有彩色顯示螢光幕才能夠看到這 16 種不同的顏色。

在低解析度圖形狀況下，螢幕的底端將留有 4 列的文字資料行；但是您可以在 GR 命令之後使用 POKE -16302，0 這個命令消除了這 4 列文字資料的顯示位置，使您的圖形顯示幕擴增到 40 行乘上 48 列的大小。

您也可以使用 TEXT 及 HOME 或 CALL -936 命令離開低解析度圖形狀況，返回到顯示文字資料的狀況。

注意：請參考 APPLESOFT][Tutorial 及 APPLESOFT][Manual 中對繪圖指令的描述。

```
10  TEXT : HOME : CLEAR : POKE 216,0:A =  FRE (0)
90  VTAB 22: PRINT  TAB( 10)"TYPE ANY KEY TO QUIT"
100  GR
110  FOR I = 1 TO 50
120 C =  INT ( RND (1) * 16)
130  COLOR= C
140 X =  INT ( RND (1) * 20)
150 Y =  INT ( RND (1) * 20)
160  PLOT X,Y: PLOT X,39 - Y
170  PLOT 39 - X,Y: PLOT 39 - X,39 - Y
180  NEXT I
190  FOR J = 1 TO 1500: NEXT J
200 K =  PEEK ( - 16384): IF K < 127 THEN 100
210  POKE  - 16368,0
220  TEXT : HOME : VTAB 10
230  PRINT "GOODBYE FOR NOW..."
240  END
```

圖9 -2 另外一個低解析度圖形的例子

在進入高解析度圖形的範疇之前，我們先提供兩個低解析度圖形的例子供您參考；請參看圖9 － 2及9 － 3 。

圖9 － 2將在4個區域內隨機的繪出彩色的"點"；因此，您將在螢光幕上看到萬花筒一般的效果；每當一個新的型式開始時就會改變前面所繪出的圖形。

您必須使用CTRL C打斷這個程式並且使用TEXT 命令返回到顯示文字資料的情況；您可以試試看在程式中加入一些測試鍵盤的指令使得這個程式能夠很簡潔地結束，我們鼓勵您這麼做。

下一個程式就是稱爲 Fifteen puzzle 的遊戲程式；在這個程式中您將看到具有4列及4行的16個元素，其中的15個元素都有它的值，1－9及A－F，隨意地散佈在這個方塊中，如果您接受這樣的安排，那麼您就必須將它們重新地像下面一般地排列好。

```
Fifteen puzzle 範例

開始          解
-----        --------
A 1 8 4      1 2 3 4
F E B 5      5 6 7 8
9 C 7        9 A B C
D 2 3 6      D E F      <--  空 白
```

重新安排位置的工作必須移動唯一的空白到其他的位置，並且用其他的元素來代替這個遺留的位置，但是每次只能移動一個元素。這個工作必須集中注意力、思考力並且花費許多的時間去完成。

順便一提的是，這個程式是用整數 BASIC 語言設計的，但是它可以很容易地轉換成 APPLESOFT][BASIC 的程式；這個工作就留給您來做了吧！

祝好運！

```
>LIST

  1 REM *** FIFTEEN PUZZLE ***
  2 REM *
  3 REM * WRITTEN BY: JL CAMPBELL
  4 REM *       DATED: 08/30/78
  5 REM *
  6 REM * COMPUTER: APPLE II
  7 REM * LANGUAGE: INT. BASIC
  8 REM *
  9 REM **********************
 10 CALL -936:PRINT:TAB 8:PRINT"*** FIFTEEN
     PUZZLE ***":PRINT
 22 PRINT "THIS PUZZLE HAS 4 ROWS & COLUMNS."
 24 PRINT "THERE IS ONLY ONE EMPTY SPACE. THE"
 26 PRINT "OBJECT IS TO ARRANGE ALL
     THE TILES INTO"
 28 PRINT "ASCENDING ORDER OF NUMBERS AND THEN"
 30 PRINT "LETTERS FROM LEFT TO RIGHT."
 32 PRINT "YOU ARE ALLOWED TO MOVE ONLY ONE AT A"
 34 PRINT "TIME, HORIZONTALLY OR VERTICALLY."
```

```
  36 PRINT:PRINT" I WILL MAKE SURE YOU DO NOT
CHEAT!!"
  38 PRINT:PRINT "WHEN ASKED ENTER ROW
     S FIRST(1-4), THEN"
  40 PRINT "ENTER THE COLUMNS(1-4) NEX
     T SEPARATED"
  42 PRINT "BY A COMMA.  THEN HIT RETURN."
  44 PRINT:PRINT "PLACE YOUR BRAIN IN
     GEAR!!! GOOD LUCK!!
  46 PRINT:PRINT
  90 DIM A(16),Z(16),ANS$(3)
 100 FOR I = 1 TO 15: A(I) = I: Z(I) = 0: NEXT I
 110 A(16)=0: Z(16)=0
 120 FOR I - 1 TO 15
 130 N = RND(16) + 1
 140 IF Z(N) < > 0 THEN 130
 150 Z(N) = A(I)
 160 NEXT I
 170 INPUT "HIT RETURN TO START PLAYING.",ANS$
 180 GOSUB 1100
 200 GOSUB 1000
 215 ER = 0: E = 1: CALL -936
 220 VTAB 21: INPUT "ROW,COL OF TILE T
     O BE MOVED.  ",R,C
 230 VTAB 22: INPUT "ROW,COL OF TILES
     NEW POSITION ",I,J
 240 IF ABS(1 - R) + ABS(J - C) < 2 THEN 300
 250 CALL -936: VTAB 23: PRINT "NOW, YO
     U KNOW THAT ISN'T A LEGAL MOVE."
 260 PRINT "THAT IS MISTAKE NUMBER ";E
     ;". ":PRINT "PLAY HONESTLY !!!":
     FOR Q=1 TO 2500:NEXT Q
 270 E = E + 1: CALL -936: IF E > 2 THEN 290
 280 GOTO 220
 290 PRINT "I WARNED YOU!!  GOODBYE YO
     U TURKEY!":END
 300 IF Z((I - 1) * 4 + J) = 0 THEN 360
 310 PRINT "WARNING!! THAT MOVE IS NOT
     ALLOWED."
 320 GOTO 220
 330 GOTO 290
 360 Z((I - 1) * 4 + J) = Z((R - 1) * 4 + C)
 370 Z((R - 1) * 4 + C) = 0
 380 FOR K= 1 TO 16
 390 IF A(K) < > Z(K) THEN 200
 400 NEXT K
 410 CALL -936: VTAB 22: PRINT "CONGRATULATIONS!!"
 420 PRINT "YOU HAVE SOLVED THE PUZZLE
 430 END
```

```
1000 REM ** PRINT GRAPH SUBR **
1010 FOR I = 1 TO 4
1020 FOR J = 1 TO 4
1030 X = (J - 1) * 10 + 5: Y = (i - 1) * 10 + 2
1040 W = Z((i - 1) * 4 + J)
1050 GOSUB 2000 + 100 * W
1060 NEXT J,I
1070 CALL -1052: CALL -1052: CALL -1052
1080 RETURN
1100 CALL -936: GR: COLOR=0
1110 FOR K=0 TO 39:HLIN 0,39 AT K:NEXT K
1120 COLOR=9
1130 HLIN 0,39 AT 0:HLIN 0,39 AT 39:VLIN 0,39 AT
0:VLIN 0,39 AT 39
1140 FOR K=9 TO 30 STEP 10:HLIN 0,39 AT K:HLIN 0,39
AT K+1:VLIN 0,39 AT K:VLIN 0,39 AT K+1:NEXT K
1150 RETURN
2000 COLOR = 0
2010 FOR IDX=0 TO 6
2020 HLIN X-3,X+3 AT Y+IDX
2030 NEXT IDX
2040 RETURN
2100 COLOR=1
2110 VLIN Y,Y+6 AT X: PLOT X-1,Y+1:HLIN X-2,X-1 AT
Y+2:HLIN X-2,X+2 AT Y+6
2120 RETURN
2200 COLOR=2
2210 HLIN X-2,X+2 AT Y+6:HLIN X-1,X+1 AT Y:PLOT
X-2,Y+1
2220 VLIN Y+1,Y+2 AT X+2:PLOT X+1,Y+3:PLOT
X,Y+4:PLOT X-1,Y+5
2230 RETURN
2300 COLOR=3
2310 HLIN X-2,X+2 AT Y:HLIN X,X+2 AT Y+3:HLIN
X-1,X+1 AT Y+6
2320 PLOT X+2,Y+1:PLOT X+1,Y+2:VLIN Y+4,Y+5 AT
X+2:PLOT X-2,Y+5
2330 RETURN
2400 COLOR=4
2410 VLIN Y,Y+6 AT X+1:HLIN X-2,X+2 AT Y+3
2420 PLOT X,Y:PLOT X-1,Y+1:PLOT X-2,Y+2
2430 RETURN
2500 COLOR=5
2510 HLIN X-2,X+1 AT Y:HLIN X-2,X+2 AT X+3:HLIN
X-1,X+1 AT Y+6
2520 VLIN Y,Y+3 AT X-2:VLIN Y+3,Y+5 AT X+2:PLOT
X-2,Y+5
2530 RETURN
```

```
2600 COLOR=6
2610 HLIN X-2,X AT Y:HLIN X-2,X+2 AT Y+3:HLIN
X-2,X+2 AT Y+6
2620 VLIN Y,Y+6 AT X-2:VLIN Y+3,Y+6 AT X+2
2630 RETURN
2700 COLOR=7
2710 HLIN X-2,X+2 AT Y
2720 VLIN Y,Y+2 AT X+2:VLIN Y+3,Y+4 AT X+1:VLIN
Y+5,Y+6 AT X
2730 RETURN
2800 COLOR=8
2810 HLIN X-2,X+2 AT Y:HLIN X-2,X+2 AT Y+3:HLIN
X-2,X+2 AT Y+6
2820 VLIN Y,Y+6 AT X-2:VLIN Y,Y+6 AT X+2
2830 RETURN
2900 COLOR=9
2910 HLIN X-2,X+1 AT Y:HLIN X-2,X+1 AT Y+3:HLIN
X-1,X+1 AT Y+6
2920 VLIN Y,Y+6 AT X+1:VLIN Y,Y+3 AT X-2
2930 RETURN
3000 COLOR=10
3010 HLIN X-1,X+1 AT Y:HLIN X-2,X+2 AT Y+3
3020 VLIN Y+1,Y+6 AT X-2:VLIN Y+1,Y+6 AT X+2
3030 RETURN
3100 COLOr=11
3110 HLIN X-2,X+1 AT Y:HLIN X-2,X+1 AT Y+3:HLIN
X-2,X+1 AT Y+6
3120 VLIN Y,Y+6 AT X-2:VLIN Y,Y+6 AT X+1:VLIN
Y+1,Y+2 AT X+2:VLIN Y+4,Y+5 AT X+2
3130 RETURN
3200 COLOR=12
3210 HLIN X-1,X+1 AT Y:HLIN X-2,X+2 AT Y+6
3220 VLIN Y,Y+6 AT X-2:PLOT X+1,Y+1:PLOT X+1,Y+5
3230 RETURN
3300 COLOR=13
3310 HLIN X-2,X+1 AT Y:HLIN X-2,X+2 AT Y+6
3320 VLIN Y,Y+6 AT X-2:VLIN Y,Y+6 AT X+1:VLIN
Y+1,Y+5 AT X+2
3330 RETURN
3400 COLOR=14
3410 HLIN X-2,X+1 AT Y:HLIN X-2,X+1 AT Y+3:HLIN
X-2,X+2 AT Y+6
3420 VLIN Y,Y+6 AT X-2
3430 RETURN
3500 COLOR=15
3510 HLIN X-2,X+1 AT Y:HLIN X-2,X AT Y+3
3520 VLIN Y,Y+6 AT X-2
3530 RETURN
```

圖9-3 fifteen puzzle程式

9.2 高解析度圖形

高解析度的圖形因爲具有更精細的圖形顯示能力，因此大都被使用在較細緻的遊戲以及商業程式的圖形顯示上。至於更精密的圖形顯示您可以參考下一節中所討論的方法。

您使用 HGR 或是 HGR2 命令進入高解析度圖形情況，這兩個命令分別將螢幕的顯示劃分爲 280（行）乘上 160（列）以及 280（行）乘上 192（列）個顯示位置。行的範圍是由 0 到 279，而列則是由 0 到 159 或是 191；位置的計算是由螢幕左上角的地方開始的。如您所見到的，您可以發現在螢幕上一個高解析度點較低解析度的點要小的多了。因此，我們稱這種顯示情形爲高解析度圖形。

您使用 HGR 命令所建立顯示幕的下方仍然留有 4 行的文字資料行，但是這種情形您可以在 HGR 命令之後加上 POKE -16302,0 命令使得螢幕的顯示成爲 280 行乘上 192 列的大小。

在繪圖的時候，我們使用 HCOLOR 命令設定圖型或是點的顏色，這些顏色您可以參看附錄A。但是在高解析度圖形情況下您所能使用的顏色只有 8 種。

```
10 HGR
20 HCOLOR = 3
30 HPLOT 0,0 TO 279,0
40 HPLOT 279,0 TO 279,159
50 HPLOT 279,159 TO 0,159
60 HPLOT 0,159 TO 0,0
70 END
```

圖 9 - 4 高解析度圖形例子

在圖 9 － 4 中的程式也會延著螢光幕繪出一個方框框；請您注意這個方框框與圖 9 － 1 中所繪出的框框，它們之間的不同。

TEXT 命令如同在低解析度圖形中所產生的功用一般，使您離開高解析度圖形狀況而返回到文字資料的狀況。在我們所討論的範例程式中所有的圖

形都將留在螢光幕上，您只有打入 TEXT 命令將它們移開。

```
1  REM  **** HIRES DEMO ****
2  REM  *
3  REM  * WRITTEN BY:JL CAMPBELL
4  REM  *      DATED:11/04/1979
5  REM  *
6  REM  * COMPUTER:APPLE II
7  REM  * LANGUAGE:APPLESOFT II
8  REM  *
9  REM  *******************
10  DIM XP(14),YP(8)
20  TEXT
30  FOR I = 1 TO 14: READ XP(I): NEXT I
35  DATA
0,20,40,60,80,100,120,140,160,180,200,220,240,260
40  FOR I = 1 TO 8: READ YP(I): NEXT I
45  DATA  0,20,40,60,80,100,120,140
90  GOTO 3000
100  IF Y% >  = 140 AND (MV% = 4 OR MV% = 5 OR MV% = 7)
THEN Y% = 0
110  FOR I = 0 TO 19
120  HPLOT X% + I,Y%
130  NEXT I
140 X% = X% + 20
150  RETURN
200  REM  * GO LEFT SUBR
210  FOR I = 0 TO 19
220  HPLOT X% - I,Y%
230  NEXT I
240 X% = X% - 20
250  RETURN
300  REM  * GO UP SUBR
310  FOR I = 0 TO 19
320  HPLOT X%,Y% - I
330  NEXT I
340 Y% = Y% - 20
350  RETURN
400  REM  * GO DOWN SUBR
410  FOR I = 0 TO 19
420  HPLOT X%,Y% + I
430  NEXT I
440 Y% = Y% + 20
450  RETURN
500  REM  * RT DIAG DN SUBR
510  FOR I = 0 TO 19
520  HPLOT X% + I,Y% + I
530  NEXT I
```

```
540 Y% = Y% + 20:X% = X% + 20
550  RETURN
600  REM  * RT DIAG UP SUBR
610  FOR I = 0 TO 19
620  HPLOT X% + I,Y% - I
630  NEXT I
640 Y% = Y% - 20:X% = X% + 20
650  RETURN
700  REM  * LT DIAG DN SUBR
710  FOR I = 0 TO 19
720  HPLOT X% - I,Y% + I
730  NEXT I
740 Y% = Y% + 20:X% = X% - 20
750  RETURN
800  REM  * LT DIAG UP SUBR
810  FOR I = 0 TO 19
820  HPLOT X% - I,Y% - I
830  NEXT I
840 Y% = Y% - 20:X% = X% - 20
850  RETURN
900  REM  * KEYBOARD TEST SUBR
910  VTAB 21: HTAB 5: PRINT "< TYPE ANY KEY TO STOP
DEMO >"
920 SW = 0
930  IF  PEEK ( - 16384) > 127 THEN SW = 1
940  POKE  - 16368,0
950  POKE  - 16388,0
960  RETURN
1000  HGR : REM  * RANDOM LINES
1010 X% =  RND (1) * 280
1020 Y% =  RND (1) * 160
1025  HCOLOR=  INT ( RND (1) * 7 + 1)
1030  HPLOT X%,Y%
1040  REM  * PLOT LINE
1060 X% =  RND (1) * 280
1070 Y% =  RND (1) * 160
1080  HPLOT  TO X%,Y%
1090  GOSUB 900
1100  IF SW = 1 THEN  RETURN
1110  GOTO 1040
1500  HGR
1510 X% =  RND (1) * 280
1520 Y% =  RND (1) * 160
1530 ST% =  RND (1) * 10 + 1
1540  HCOLOR=  INT ( RND (1) * 8)
1550  VTAB 23: CALL  - 958: PRINT  TAB( 12)"< STEPPING
BY ";ST%;" >"
1560  FOR J = 0 TO 159 STEP ST%
1570  HPLOT X%,Y% TO 0,J
1580  NEXT J
```

```
1590  FOR J = 0 TO 279 STEP ST%
1600  HPLOT X%,Y% TO J,159
1610  NEXT J
1620  FOR J = 159 TO 0 STEP  - ST%
1630  HPLOT X%,Y% TO 279,J
1640  NEXT J
1650  FOR J = 279 TO 0 STEP  - ST%
1660  HPLOT X%,Y% TO J,0
1670  NEXT J
1680  FOR I = 1 TO 1500: NEXT I
1880  GOSUB 900
1890  IF SW = 1 THEN  RETURN
1900  GOTO 1510
2000  HGR : REM  * ROTATING SQUARES
2010 ST% =  RND (1) * 4 + 2
2020 X% = 0:Y% = 0
2030 DX% =  RND (1) * 280:DY% =  RND (1) * 160
2040  HCOLOR=  INT ( RND (1) * 8)
2050  FOR I = 1 TO 50
2060  HPLOT DX%,Y% TO 279,DY% TO 279 - DX%,159 TO 0,159
- DY% TO DX%,0
2070 DX% = DX% + ST%:DY% = DY% + ST%
2080  IF DX% >  = 277 THEN DX% = 0
2090  IF DY% >  = 157 THEN DY% = 0
2100  NEXT I
2110  GOSUB 900
2120  IF SW = 1 THEN  RETURN
2130  GOTO 2010
2400  RETURN
2500  HGR : REM  * MADNESS DRIVER
2510 X% = XP( RND (1) * 14 + 1)
2520 Y% = YP( RND (1) * 8 + 1)
2530  REM
2540 R% =  RND (1) * 10 + 1
2550  HCOLOR=  INT ( RND (1) * 8)
2560  FOR J = 0 TO R%
2570 MV% =  RND (1) * 8 + 1
2580  IF X% >  = 260 AND (MV% = 1 OR MV% = 5 OR MV% =
6) THEN X% = 0
2590  IF X% <  = 20 AND (MV% = 2 OR MV% = 7 OR MV% = 8)
THEN X% = 279
2600  IF Y% >  = 160 AND (MV% = 4 OR MV% = 5 OR MV% =
7) THEN Y% = 0
2610  IF Y% <  = 20 AND (MV% = 3 OR MV% = 6 OR MV% = 8)
THEN Y% = 159
2620  ON MV% GOSUB 100,200,300,400,500,600,700,800
2700  NEXT J
2710  GOSUB 900
2720  IF SW = 1 THEN  RETURN
2730  GOTO 2530
```

```
3000  HOME : PRINT : PRINT  TAB( 9)"***** HIRES DEMO
*****": PRINT
3010  PRINT  TAB( 5)"1- RANDOM LINES": PRINT
3020  PRINT  TAB( 5)"2- RADIATING LINES": PRINT
3030  PRINT  TAB( 5)"3- ROTATING SQUARES": PRINT
3040  PRINT  TAB( 5)"4- MADNESS": PRINT
3045  PRINT  TAB( 5)"5- SUPERROSES": PRINT
3050  PRINT  TAB( 5)"6- LIMACON SURFACE": PRINT
3055  PRINT  TAB( 5)"7- EXIT PROGRAM": PRINT
3060  HTAB (5): INPUT "WHICH ONE? ";Z
3070  IF Z >  = 0 AND Z <  = 7 THEN 3100
3080  PRINT : PRINT "INCORRECT ENTRY....TRY AGAIN": FOR
I = 1 TO 1500: NEXT I: GOTO 3000
3100  ON Z GOSUB 1000,1500,2000,2500,4000,4500,5000
3110  TEXT : GOTO 3000
4000  TEXT : REM  ROSES SUBR
4010  HOME : PRINT : PRINT  TAB( 10)"** ROSES **":
PRINT
4020  VTAB (10): INPUT "HOW MANY LEAVES? ";L%
4030  HGR :ST = 0
4035  HCOLOR= 3
4040  FOR I = 0 TO 360
4050 R = 79 *  SIN (L% * ST)
4052 R1 = 79 *  SIN (L% * ST + 1.57)
4054 R2 = 79 *  SIN (L% * ST + 3.14)
4056 R3 = 79 *  SIN (L% * ST + 4.72)
4060 X% = 1.5 * R *  COS (ST)
4062 X1% = 1.5 * R1 *  COS (ST)
4064 X2% = 1.5 * R2 *  COS (ST)
4066 X3% = 1.5 * R3 *  COS (ST)
4070 Y% = R *  SIN (ST)
4072 Y1% = R1 *  SIN (ST)
4074 Y2% = R2 *  SIN (ST)
4076 Y3% = R3 *  SIN (ST)
4080  HPLOT X% + 140,Y% + 80
4082  HPLOT X1% + 140,Y1% + 80
4084  HPLOT X2% + 140,Y2% + 80
4086  HPLOT X3% + 140,Y3% + 80
4090  GOSUB 900
4100  IF SW = 1 THEN  RETURN
4110 ST = ST + 3.14159 / 360
4120  NEXT I
4130  FOR I = 1 TO 2500: NEXT I
4140  GOTO 4000
4500  REM  LIMACON SURFAFE
4510  HGR :ST = 0
4515  HCOLOR= 3
4520 A% = 25:B% = 59
4530  HPLOT 40,20 TO 40,140
4540  HPLOT 40,80 TO 240,80
```

```
4600  FOR I = 0 TO 720
4610 R = A% + B% *  COS (ST)
4620 X% = 2.0 * R *  COS (ST)
4630 Y% = R *  SIN (ST)
4640  HPLOT X% + 40,Y% + 80
4650  GOSUB 900
4660  IF SW = 1 THEN  RETURN
4670 ST = ST + 3.14159 / 360
4680  NEXT I
4690  FOR I = 1 TO 4000: NEXT I
4700  RETURN
5000  POKE  - 16298,0
5010  END
```

圖 9 - 5　高解析度圖形示範程式

這個程式說明了幾個高解析度圖形的程式段，並且將 Apple][的能力盡量地發揮了一下；同時，我們也再度地練習了檢查鍵盤輸入的方法，使用手册及副程式的例子，更重要的我們也將三角函數的功能在例子中告訴了您。

我們必須承認這個程式並不是在各方面都符合結構化程式的要求；如果您願意，我們希望您能將它改進到令人滿意的地步。

9.3　造型表 (Shape Tables)

造型表就是用來處理高解析度圖形造型的工具，我們並不能在低解析度圖形中使用它。

一個造型可以使用一連串的數元組來定義，而每一個數元組都分成三段，每一段都存著一個繪圖係數（ plotting vector ）用來繪出這個造型中的一個點。在您定義完了造型之後，這些數元組就要被轉換成十六進位的值，然後使用監督系統的功能將這些值輸入到記憶體內。我們在這裏並不討論如何建立一個造型表的方法，詳細的情形您可以參考由編者所譯，儒林圖書公司出版的“ APPLE][使用手册”一書。

在您定義完了造型表的所有係數碼之後，您可以使用幾個命令使您設計的造型顯示在螢光幕上。圖 9 － 6 就是一個範例的程式。

```
1   REM ***** SHAPE DEMOS *****
2   REM *
3   REM * WRITTEN BY:JL CAMPBELL
4   REM *       DATED:06/12/80
5   REM *
6   REM * LANGUAGE:APPLESOFT II
7   REM * COMPUTER:APPLE II
8   REM *
9   REM ***********************
10   TEXT : CLEAR : POKE 216,0:A =  FRE (0)
30   GOTO 1000
50   REM  RANDOM COLOR
52 C =  INT ( RND (1) * 8)
54   IF C = OC THEN 52
56   HCOLOR= C:OC = C
59   RETURN
69   REM * TEST KEYBOARD
70 SW = 0
74   IF  PEEK ( - 16384) > 127 THEN SW = 1
76   POKE  - 16368,0
78   RETURN
99   REM * FLASHING SCREEN
100   GOSUB 50: HPLOT 0,0: CALL 62454
105   GOSUB 70
110   IF SW = 1 THEN  RETURN
115   GOTO 100
199   REM * DRAW SHAPE
200 ST =  - ST: GOSUB 50
205 B = 1:E = R
210   IF ST < 1 THEN B = R:E = 1
230   FOR I = B TO E STEP ST
235   ROT= I: SCALE= I
240   DRAW SH AT X,Y
245   NEXT I
250   GOSUB 70
255   IF SW = 1 THEN  RETURN
260   CALL 62454: GOTO 200
999   REM * LOAD SHAPE TABLE START ADDRESS
1000   POKE 232,0: POKE 233,3
1010   REM * LOAD SHAPE TABLE DATA
1020   FOR LOC = 768 TO 783
1030   READ BYTE: POKE LOC,BYTE
1040   NEXT LOC
1050   DATA 2,0,6,0,9,0,44,62,0,
        44,46,62,62,60,44,0
1060   ROT= 0: SCALE= 5:X = 139:ST = 1
1999   REM * MAIN DRIVER
```

```
2000  TEXT : HOME : PRINT : PRINT  TAB( 10)"** HIRES
       GRAPHICS **": PRINT
2002  PRINT "MENU:": PRINT "-----"
2004  PRINT  TAB( 5)"1- SQUARE": PRINT
2006  PRINT  TAB( 5)"2- CROSS ": PRINT
2008  PRINT  TAB( 5)"3- FLASHING": PRINT
2010  PRINT  TAB( 5)"0- EXIT PROGRAM"
2020  VTAB (14): CALL  - 958: INPUT "WHICH ONE? = ";N:
       SH = N
2025  IF N = 0 THEN  VTAB 4: CALL  - 958: VTAB 12:
       PRINT "GOODBYE FOR NOW....": END
2030  IF N < 1 OR N > 3 THEN  PRINT  CHR$ (7):
       GOTO 2020
2040  HGR : POKE  - 16302,0
2050  CALL 62454
2060  IF SH = 1 THEN Y = 95:R = 80
2070  IF SH = 2 THEN Y = 110:R = 48
2080  ON N GOSUB 200,200,100
2090  GOTO 2000
```

圖 9-6 造型表示範程式

這個程式將把程式中定義的造型（在行號 1050 定義的）顯示在高解析度顯示幕上；當脚註值（index）I 增加的時候，這個造型將改變它的大小並做旋轉。

行 號	描 述
10	—設定乾淨的機器。
50－59	—選定新的 HCOLOR 值。
69－78	—測試鍵盤。
99－115	—將螢幕用上一次的 HCOLOR 填滿。
199－260	—定造型的大小，繪出並且旋轉它。
	—定義負的增加值，ST 。
	—選擇 HCOLOR 值，GOSUB 50。
	—定義 FOR...NEXT 廻圈中的起始值 B 及終止值 E。
	—依照 I 值旋轉造型。
	—依照 I 值定造型的大小。

	一在 X,Y 位置繪出造型 SH 。
	一測試鍵盤，GOSUB 70 。
	一將螢幕依照上一次的HCOLOR填滿。
1000－1050	一將造型表存入記憶體內。
	一使用的是1050行號DATA內的資料。
1060－	一設定 ROT 值。
	一設定 SCALE 值。
	一設定繪圖的X位置值。
	一設定 ST 值為1 。
2000－2030	一主程式段。
2040－	一設定圖形狀況。
2050－	一按照上一次的HCOLOR將螢幕填滿。
2060－2070	一設定繪圖的Y位置值。
	一設定繪圖範圍，R 值。
2080－	一執行選定的副程式。
2090－	一返回到主手册中。

9.4 將圖形印製在紙上

將圖形以黑白的顏色印製在紙上的工作，可以說是這幾年來列表機的一個新發展（且前已有將彩色圖形印在紙上的列表機了）；下面所要討論的就是如何利用列表機將圖形印在紙上的問題。

做這項工作的第一個要求就是我們必須將顯示在螢光幕上的圖形存到某種儲存設備上去；一般情況下我們使用的都是磁碟機，這樣子我們才可以在需要的時候將這個圖形重新存回記憶體內；至於儲存的方法我們已經在第八章中討論過了。

至於下一個要求（也許是最重要的條件）就是您必須具備能夠繪圖的列表機。至於具有這種功能的列表機一般都分為兩大類，它們分別是：

1. 非撞擊式的 (Non-impact)
   * 一熱感應式的 (Thermal printing)
   * 一電熱式的 (Electrosensitive printing)
   * 一電針式的 (Electrostatic printing)
   * 一噴墨式的 (Jet ink laser printing)
2. 撞擊式的 (Impact)
   * 一點矩陣式的 (Dot matrix printing)
   * 一字型的 (Shaped character printing)

非撞擊式的列表機都是使用能對熱或電熱產生感應的紙，這些紙的外層大都加上了特殊的藥品，使得在接受到列表機頭發出的感應時能夠產生變化。

熱感應式的列表機頭由許多的垂直電綫所組成，它能夠將電流轉換成熱力，當印字時這些熱力就將紙上的特殊外皮燒黑而形成了圖形。

電熱式的列表機則使用一種乾的藥品覆蓋在紙上，當電流通過時產生的電壓影響到這一層膜的時候，這一層膜就會被燒掉而留下了裏層的黑色部份，這個黑色部份就是我們需要的形狀。像這樣的紙一般都是在外層加上一層鉛的薄膜。

電針式的頭與電熱式的大致相同，但是這種方法使用了一個電針，電針上的電壓依照圖形或字形而改變，在紙上燒出原先的形狀。

噴墨式的列表機則有一個控制噴墨的噴頭，就由這個噴頭將墨噴在紙上形成我們所要輸出的各種形狀。像這種方式的列表機所印出來的字形或圖形，就完全依照它所具有的噴墨點的結構以及噴墨的密度而決定噴成的字型或圖形是否美觀了。

撞擊式的列表機傳送圖形的方式有兩種。其中一種就是色帶及紙與字型是連接在一起的，當印字或是圖形的時候就是印表頭隨著移動而完成。另外一種的構造則如打字機般的工作，在這種情況下就是字型打在色帶上而印出我們所要輸出的字型。

在撞擊式列表機上我們又可以依照它們具有的印表頭的不同而分爲兩類。

第一類就是點矩陣式的印表頭，它是由許多的印表針或是噴墨針所組成的，當它印表時就依照字型或是圖形而分別趨動這些頭形成我們所要的型狀

。目前這類頭上針的安排已經有了更大的進步，使得許多部份能夠被重複地印出形成較完美的字型。

第二種型式的頭則是使用事先做好的字型印字；這種型式的印表頭又有三種不同的種類：

* 一輪圈式的頭（Daisy wheel head）
* 一球式的頭（Ball head）
* 一滾筒式的頭（Cylinder head）

由於每一種型式的列表機或印表頭都有它的特性，因此在印字或是印圖時的品質或型狀都不儘相同；您最好能夠瞭解各種類型列表機的特性，以便在使用它們的時候能夠充份地發揮它們的功能。

在下面的幾節中我們將討論許多種不同型式的列表機。

9.4.1 非撞擊式的列表機

在這一節中所要討論的列表機是 Apple 電腦公司出品的 Silentype thermal printer 。這種列表機可以由使用者訂定許多種印圖的功能。

首先，在您要將圖形印到紙上之前，您必須先將您的圖形存在記憶體內或是顯示在螢光幕上；因此，您必須先設計一個繪圖的程式。我們在這裏假設現在您已經將圖形繪好存在記憶體內了。其次，我們假設您的圖形是存在高解析度圖形的第一頁上的（請參考"APPLEⅡ使用手册"中的高解析度圖形的說明），Silentype 列表機則與插在第1個擴充接點的介面與相連。那麼印出存在記憶體內圖形的程式就是圖9－7中的範例程式。

在操作 Silentype 列表機的時候，您可以使用立即式或間接式的命令。

大多數的列表機都可以瞭解跳頁（使用(PRINT CHR$(12)= CTRL L)做爲跳頁的命令）或跳行的命令。Silentype 列表機將跳頁的命令視爲跳固定格數的命令，這麼做的原因是要使得您撕紙的工作較易進行，而這些格數的設定可以在程式中控制。

行數的增加是以 0.0277 英吋爲單位的（也就是一格是 0.0277 英吋的

Applesoft 的程式

立即式情況	間接式情況
	90 D$ = CHR$(4): Q$ = CHR$(17)
]PR#1	100 PRINT D$;"PR#1"
]PRINT CHR$(17)	
	110 PRINT " "
	120 PRINT Q$
:	:
]PR#0	210 PRINT D$;"PR#0"

注意： CHR$(4) = CTRL D
CHR$(17)= CTRL Q

圖 9-7 使用 Silentype 程式

距離），因此 40 格就是大約 1 英吋的距離。這些格數可以使用下面的命令設定：

POKE -12531，格數

範例： POKE -12531,200

現在當您通知列表機一個移動的命令時，紙就會往前移動大約 5 英吋的距離。下面就是一個設定移動距離的程式段。

```
75 IF PEEK( -12506) = 183 THEN RETURN
80 PRINT CHR$(4);"PR#";SLOT: PRINT
85 PRINT CHR$(4);"PR#0"
90 RETURN
```

圖 9-9 設定列表機初步格式的程式

所有可供操作員選擇的功能都必須依照適當的次序完成；在您改變任何設定的元素之前您必須先設定列表機位在的擴充接點數目。當您想要改變元素值的時候，您必須使用不能被顯示出來的控制字（control character），這些字都被存在記憶體內供列表機的控制程式使用。

您也可以使用下面的程式段檢查您的列表機是否已經完成初步設定的工作了。

```
899 REM * FORM FEED
900 VTAB 4: CALL -958
905 INPUT "ENTER FORM FEED SIZE(1-255) ";FF
910 IF FF > 0 AND FF < 256 THEN 920
915 GOSUB 60: GOTO 900: REM ERROR TRAP
920 POKE -12531,FF
925 RETURN
```

注意：參看圖 4.7 以便瞭解 GOSUB 60 的程式段。
FF＝跳頁的大小

圖 9-8 跳頁的設定

我們在知道跳頁的工作如何完成之後，我們就可以訂定行與行間的距離了；藉著控制行與行間的距離，我們就可以訂定表、列、圖形等等的格式了；一個跳行的命令（由游標回頭而產生的），會把紙往前移動6格，而每一個字佔用的大小則是4格，而字與字之間的垂直距離則是2格；因此我們可以知道字與字之間的設定距離就是2格；我們可以使用下面的命令改變這個距離：

POKE -12530,距離

範例:POKE -12530,14

上面這個例子使得行之間的距離增加了3倍，這是因爲設定的距離是2格，再加上 12 格（代表2個空白行）。順便一提的是，如果您使用了跳行的命令（ PRINT CHR$(10) = CTRL J ），紙就會往上捲一行，但是印表頭依然呆在它原先的水平位置上，這時候您就必須使用一個游標回頭的命令（ PRINT CHR$(13) = CTRL M ）將印表頭移到這一行的最左邊位置。

下面的這個副程式段將設定三種不同大小的行與行間的空隔。

```
799 REM * LINE SPACING
800 VTAB 4: CALL -958
805 PRINT "CHOOSE:"
810 PRINT TAB(7) "1- SINGLE SPACING"
815 PRINT TAB(7) "2- DOUBLE SPACING"
820 PRINT TAB(7) "3- TRIPLE SPACING"
825 VTAB 9:CALL -958: GOSUB 55: REM INPUT SUBR
830 IF N >0 AND N < 4 THEN 840
835 GOSUB 60: GOTO 825: REM ERROR TRAP
840 LS = 6 * (N-1) + 2
845 POKE -12530,LS
850 RETURN
```

注意：參看圖 4.7中的 GOSUB 60 及GOSUB 55 的程式行

LS ＝行與行間的空隔

圖 9-10 設定行與空間空隔的程式

Silentype 列表機印表的時候可以採用兩種方式，一種就是雙向式的（ bi-directional ），一種則是單向式的（ uni-directional ）。採取雙向式印表的時候，印表的方式是由左至右再由右至左順序交換印製，這種方法對於列出文字資料或是報告都是最好的方法；至於單向式的印表則只能由左至右一個方向印製，這種方式適用於印出圖形或是您希望將資料的左邊對齊的時候使用的。這種方式的速度就較上一種來得慢了。

訂定印表方式的命令如下：

```
POKE -12529,方向
```

範例： `POKE -12529,0`

上面這個例子將印表的方式設定爲雙向式的；至於單向式印表的設定值則是 255 。下面這個副程式段就是用來設定印表方式的。

```
699 REM * PRINT DIRECTION
700 VTAB 4: CALL -958
705 PRINT "CHOOSE:"
710 PRINT TAB(7) "1- UNI-DIRECTIONAL"
715 PRINT TAB(9) "2- BI-DIRECTIONAL"
720 VTAB 8: CALL -958: GOSUB 55: REM INPUT SUBR
725 IF N = INT (N) THEN 735
730 GOSUB 60: GOTO 720: REM ERROR TRAP
735 IF N > 0 AND N < 3 THEN 735
740 GOSUB 60: GOTO 720: REM ERROR TRAP
745 POKE -12529,N - 2
750 RETURN
```

注意：參看圖 4.7 中 GOSUB 55 及GOSUB 60 的程式行

A − 1 = F F = 255（十進位）

圖 9 -11 印表方式的設定

Silentype 列表機有 8 種不同的印字深淺度，也就是 0 − 7 ，當數字越高的時候也就表示印出的字越黑。您也可以使用 POKE 命令將適當的深度值存到記憶體內而改變它的深淺。例如：

POKE -12528,深度

範例：POKE -12528,7

深淺度的設定值是 5 ；所以上面的例子將深淺度由 5 改爲 7 ，而這個深淺度將一直不變直到您重新設定它爲止。下面的副程式段就是用來設定顏色深淺度的。

```
599 REM * INTENSITY
500 VTAB 4: CALL -958
605 INPUT "WHICH INTENSITY?(1-7) ";PI
610 IF PI >= 0 AND PI < 8 THEN 620
615 GOSUB 60: GOTO 600: REM ERROR TRAP
620 POKE -12528,PI
625 RETURN
```

注意：參看圖 4.7 中 GOSUB 60 的程式行

PI = 印字深淺度

圖 9 − 12 設定印字深淺度的程式

至於左邊與右邊的界限（ margins ）也可以由您設定一個在准許範圍內的值；在大多數的情況下，您也許希望利用 Silentype 一行 80 字的功能；然而，有些時候您又希望在左右兩邊各多留一些的空位。Silentype 列表機一行最多可以容納 83 個字（ 0 － 82 ）；機器設定的左邊界限值爲 2 ，右邊界限值則爲 81 ，這使得正常情況下的一行可以容納 80 個字。至於左右兩邊的界限值您可以使用下面的 POKE 指令完成。

```
POKE -12527,右邊界限
POKE -12526,左邊界限
```

範例：
```
POKE -12527,15
POKE -12526,76
```

這個例子設定左邊的界限值爲 15 而右邊的界限值爲 76 ，這使得一行可以容納 60 個字。下面是一些您在設定界限值時必須注意的事項：

＊一右邊的界限值必須大於左邊界限值。

＊一左邊界限值必須大於零。

＊一右邊界限值必須小於 84 。

＊一行的寬度必須大於 4 。

下面這個副程式段就是用來設定界限值的。

```
499 REM * MARGINS
500 VTAB 4: CALL -958
505 INPUT "ENTER LEFT MARGIN(0-78)  ";LM
510 IF LM = INT(LM) THEN 520
515 GOSUB 60: GOTO 500: REM ERROR TRAP
520 VTAB 6: CALL -958
525 INPUT "ENTER RIGHT MARGIN(4-82) ";RM
530 IF RM = INT(RM) THEN 540
535 GOSUB 60: GOTO 520: REM ERROR TRAP
540 IF RM - LM > 3 THEN 550
545 GOSUB 60: GOTO 500
550 POKE -12527,RM
555 POKE -12526,LM
560 RETURN
```

注意：請參看圖 4.7 GOSUB 60 的程式行

RM＝右邊極限值

LM＝左邊極限值

圖 9-13 極限值的設定

您可以將任何一個高解析度圖形頁內所存的圖形印到紙上；您只需使用下面的命令就可以改換圖形頁了。

POKE -12525,圖形頁值

範例：POKE -12525,64

這個例子將代表高解析度圖形的第 2 頁的值 POKE 入記憶體內，這使得您可以將這一頁內的圖形印到紙上去；您也可以使用 POKE -12525,32 命令回到第 1 頁。注意您 POKE 的值是 32 乘上記憶體內的頁數；因此，您就可以使用這種方法將記憶體內任何一頁的內容印到列表機上去了。

最後，您還可以使用黑白相反的方式印表；這也就是說您可以將黑的地方印成白色，而白的地方印成黑色。這個工作同樣地是使用下面的命令完成的。

POKE -12524, 黑白相反

範例 **POKE -12524,0**

這個例子將把螢幕上的顯示按照黑白相反的方式印在紙上。您也可以使用 **POKE -12524,255** 命令返回到正常的狀況。

下面就是一些高解析度的圖形，我們利用 **Silentype** 列表機配合著立即執行的一些命令繪出到紙上的。

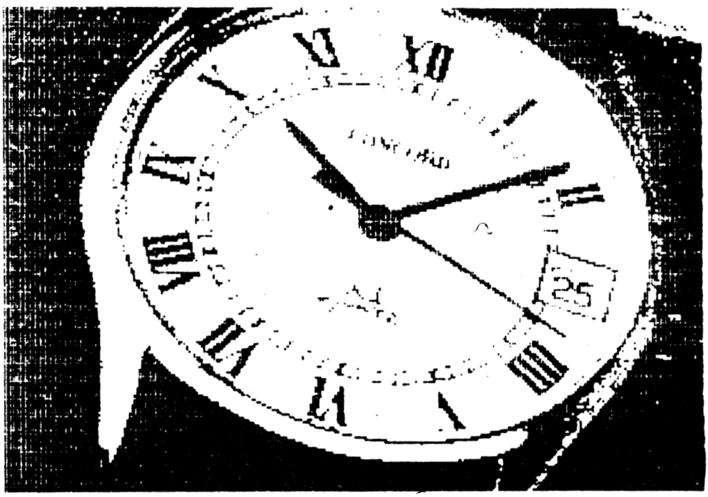
```
]POKE -12527,2

]POKE -12528,6

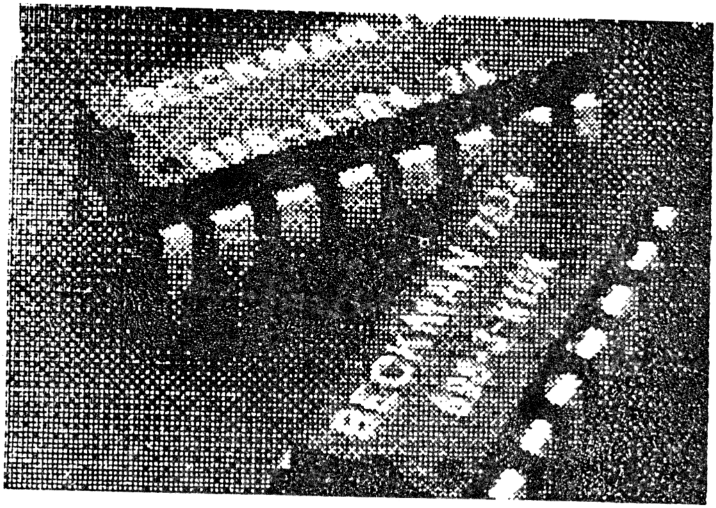
]POKE  -12529,255

]BLOAD THE TIME MACHINE.PIC
]POKE -12527,20

]PRINT CHR$(17)
```



```
]BLOAD DIP CHIPS.PIC
]PRINT CHR$(17)
```



```
]POKE -12525,64

]BLOAD WINSTON CHURCHILL.PIC
]PRINT

]PRINT CHR$(17)
```



9.4.2 撞擊式的列表機

我們在這一節中將介紹許多種撞擊式的列表機；然而，我們將選擇一個點矩陣式的及一個字型式的列表機做說明；它們分別是 Epson 的 MX 系列列表機，以及 Qume 的 Sprint 5 及 9 系列。

9.4.2.1 點矩陣式的列表機

我們所要討論的點矩陣式的列表機是 Epson MX 80 系列的列表機，這種列表機是以平行的輸出介面與電腦聯繫的。

我們討論這種列表機的原因是因爲它的普徧性，以及它能夠利用點矩陣的方式印出圖形的能力。

由於這種列表機的介面裝置是平行輸出介面卡，因此在圖 3－12 中的例子就可以用來使它印出程式列或是報表了。同時我們也可以使用一些前面所描述的控制字來控制它的作業。

至於其他的控制字，像是 CTRL N, CHR$(14) 將使您能夠得到放大的字體，CTRL O, CHR$(15) 則使得字型被縮小後印出。至於加上了圖形能力的 MX-80 及本身已經具備繪圖能力的 MX-70 ，則可以使用額外的一些控制方法使得圖形能夠被印出。

在圖形的狀況下，繪出的圖形是依照一個一個的數元而被印到紙上的；這是將圖形以數元的型式代表，然後用軟體程式控制將這些數元所代表的位置印出點來組合而成的。至於印出圖形的控制碼對於不同型式或是不同生產地的機器都有所不同，故而您必須參閱某個工廠生產這個機器時的說明手册。

在 MX-70 機器上使用繪圖的能力，您必須使用 ESC K+n1+n2 的命令，注意“ + ”號在實際的命令中並不存在，我們用在這裏只是供做說明用的。n1 及 n2 都是十六進位數字，它們被用來設定代表圖形的數元資料的數量；n1 代表低次位數字而 n2 則代表高次位數字，利用數元繪圖所能表示的點數目不能超過 480 ，因此n1及 n2 就不能超過 480 了。在印完圖形之後，隨後的資料就被視爲文字資料，而這個列表機在一行內印完了 480 個點

的位置之後就自動地返回到文字資料狀況。

9.4.2.2 字型的列表機

Qume Sprint 5 列表機是一個具有高品質字型的列表機，它可以使用平行的或串列的介面卡與電腦聯繫。它印字的速度大約是在每秒 45 到 55 個字之間，實際的速度則依賴一行中印字的數目而決定。下面是其他的一些特性：

* —可以由使用者控制印字的顏色深淺。
* —可以使用單張或連續的紙，紙上可以有固定紙的連續洞也可以沒有。
* — 96 個字的格式，可以有10，12，15 斜度的選擇。
* —每行印132 個字；依照10 字/英吋的格式。
* —每行印158 個字；依照12 字/英吋的格式。
* — 10 / 12 斜度的設定（可以由操作員重設）。
* —電動式的快速移動位置。
* — 8/48″的行間的距離（每英吋6 行）。
* — 1/48″到159/48″的距離可以使用命令設定。
* —具有5 英吋垂直迴旋的能力。
* —兩種繪圖能力（解析度可達5760 ）。
* —使用者可以控制水平、垂直位置、以及跳行跳頁等等的工作。

Sprint 5 具有的印圖能力會使得任何的空位命令或是回頭命令被取消，而只做1/60″或1/120″ 的移動。進入圖形狀況的命令是ESC 3或是 ESC G ； ESC 3 命令使得圖形是以1/60″的移動狀況印出的，至於 ESC G 則使用的是1/120″ 的狀況。至於停止印圖的動作則可以使用ESC 4 命令或是游標回頭命令。

Sprint 5可以由上往下的印表或是依相反的方向印字；使用ESC 6 命令就可以使印字的方向變爲由右至左，取消的方式則是使用 ESC 5 或是游標回頭命令。

下面是一些Sprint 5列表機的命令：

* 一 SPACE (SP)；這個命令的值是依賴空格指標的設定值而決定的；空格指標的值可以是每英吋12個字或是10個字。
* 一 BACK SPACE (BS)；這個命令的值必須依照空格大小值的設定而決定。
* 一游標回頭 (CR)；這個命令使讀寫頭回到左邊的界限，並且結束圖形或是回頭印表的工作。
* 一跳行 (LF)；當列表機開關被打開，或者使用命令都可以使紙往上移動1/6″；至於在印圖形時則往上移動1/48″。
* 一往上跳行(ESC LF)；與往上移動的方式相同，只是方向相反而已。
* 一跳半行 (ESC U)；會使紙往上移動的距離減半。這個命令不能在繪圖時使用。
* 一往上跳半行(ESC D)；與跳半行的方式相同，只是方向相反而已。
* 一平行移動位置的設定 (HT)；使印表頭移到最近的設定位置去。
* 一跳頁 (FF)；使紙往上移到下一頁的表頭爲止，移動的大小則依照目前印表頭所在的位置以及頁的大小而定；印表機上的按鍵也可以做這個工作。
* 一(ESC SP)；會在印字輪(printwheel)的004位置上印上一個空白或是一個字。
* 一(ESC /)；會在印字輪上(printwheel)的002位置上印上一個空白或是一個字。
* 一取消輸入(ESC X)；如果這個命令不是一個二進位資料的命令，則取消剛輸入的這個命令。

Sprint 5使用下面的命令設定左右界限(margins)及停留位置(tabs)：

* 一 (ESC 1)；在目前印表頭所在的位置設定一個停留位置。
* 一 (ESC C1,C0.)；不管現在印表頭所在的位置而設定一個或多個的停留位置。這些停留位置可以由 C1 及 C0 所代表的行數所設定，行與行之間必須加上逗號，並且在結尾加上句點；C1 的值可以由 0 到 F (0 到 15)，C2 則可以由 0 到 9 。

  ESC b2,c6,d0 這個命令將停留位置設定在第 112，126 及150 行的地方。
* 一 (ESC 8)；將去除目前印表頭所在的停留位置。
* 一 (ESC C1, C0, C1.)；將去除一個或多個的停留位置(不論目前印表頭在什麼地方)，C1 及 C0 值的設定與上面所描述的方式相同。
* 一 (ESC 2) ；將去除所有停留位置的設定。
* 一 (ESC 9)；將把左邊的界限設定爲目前印表頭所在的位置。
* 一 (ESC 0) ；將把右邊的界限設定在目前印表頭所在的位置。

設定列表機的命令如下：

* 一 (ESC B)；這個命令將使得具有兩色的色帶跳到印字的位置，這使得印出的字是打在兩色色帶的黑色部份上的。
* 一 (ESC A)；這會使得色帶往下落一半，印出的字將打在紅色的部份；直到您使用了 ESC B 命令取消前面的命令後色帶才會跳上原來的位置。
* 一 (ESC F d1 d2) ；設定一頁的長度(以行爲單位)，最多可以定到一頁 127 行(1 行1/6″)。D1 値可以由 0 到C (0 到 12)，D2 值則可以由 0 到 9 。
* 一 (ESC N)；對下一個輸入的字防止印表頭的移動。
* 一 (ESC L d1 d2)；設定行與行之間的寬度，這個值是以 1/48″爲單位的；D1 可以由 0 到F，而 d2 則可以由 0 到 9。
* 一 (ESC SUB I)；會使得列表機回到起始的狀況，任何剛被接收的命令都將不被執行。

您使用以上介紹的命令就可以適當的應用列表機的能力了；至於Sprint 9的列表機也可以印出圖形，它具有的功能與Sprint 5大致差不多。

使用字型式的列表機印出圖形的第一步就是您必須學會如何將高解析度圖形頁的位置找出來，其次您必須將這個圖形利用某個符號或是字印到紙上；這就是說您必須在記憶體內將存有圖形的數元組找出，並將它們以數元型式使用字或符號印到紙上。當您印圖的時候，您必須控制印表的水平及垂直位置，同時您也要選擇您要使用來組合圖形的字或符號。

9.5 總　結

在這一章中我們討論了Apple][繪圖的能力；第一節討論的是低解析度及高解析度螢光幕的組成型式，同時我們也對文字資料幕的合併使用做了介紹。

圖形，尤其是彩色圖形的重要性日益增加，尤其在目前我們發現使用圖形來表示訊息的方式對於許多方面的影響力都是很大的，不論在教育方面，實驗工作，或是設計製圖等等都有不少的貢獻。至於在商業方面及娛樂事業也都發現了它的實用性，我們現在已經可以在市面上看到許多的商業或是遊戲方面的圖形程式了。

我們對於高解析度及低解析度圖形都做了說明及舉了適當的例子，並且對操作它們的指令也都做了描述。然後，我們見到的是造型表的一個範例程式，程式中對於DRAW ，ROT及 SCALE 命令也都有了明確的討論。

最後我們討論的是印圖的問題，對於每一種可以印圖的列表機的特性及使用方法我們都做了說明，甚至給了您一些印圖的例子以及操作的程式段。

第10章

將程式做好完整的記錄

10.0 大 綱

記錄的工作往往讓人覺得無趣，但是對於設計程式的工作而言卻是非常有用的，雖然它對於利用程式解決問題並沒有太大的幫助，但是卻能夠在往後節省您許多的時間與精力。圖10－1中列舉了許多做程式記錄時需要注意的事項。

1. 所有的記錄都必須有標題與識別。
2. 必須包含著問題的敍述。
3. 所有的複製資料都必須有相同的隱秘性。
4. 對於所有的修正工作都必須有完整的說明。
5. 特殊的符號都必須加以說明。
6. 確實修改程式的日期應該加以註明。

圖10-1 程式記錄的注意事項

下面有幾種記錄的型式是我們所要討論的；它們是：

1. 分析式的（Analytical）
2. 系統的（Systems）
3. 程式的（Program）

4. 幫助操作及使用者的（Operations/User Aid ）

10.1　分析式的記錄工作

分析式的記錄工作主要著眼於整個計劃的每一項枝節，它包含了這個計劃中所有要使用的記錄及報表。由於這本書的目的並不在專注於討論計劃的問題，因此我們對這種方式的記錄工作也不會討論的非常詳細，但是我們將把分析式記錄中的重要項目說明一下，因為它們在以後的幾節中都將被使用到。

首先，我們必須寫一個關於計劃的總論，在這個總論中我們必須說明計劃的目的及產生的功能，同時包含著所有的輸出與輸入的檔。

總論中也必須包含著系統的流程圖以及說明的部份，對於整個系統的流程以及計劃的定義，程式的大小等等問題也要有詳細的描述。

在總論之後，就是說明計劃中每一個檔的部份，這一部份中對於檔、記錄及資料的格式都必須有較詳盡的描述，並且對於檔的內容、修改等等情形也必須顯示出來。

當討論到輸入問題的時候，您應該將所有輸入的目的及代表的意義說明清楚；同時您也應該加上一個異動表，裏面記載著變數的意義，對程式所發生的影響，以及出現的頻率等等資料。

這一段結束的部份應該包含著檔的規格，及一些實際的例子。

就如同輸入部份一般，輸出的部份也必須包含著對輸出的標識及目的的解說；同時也應該包含著異動表，對於輸出的各種項目做詳細的說明。

至於處理部份的描述則必須能夠顯示出輸入資料對於整個程序及輸出結果的影響情形。

測試部份的描述則應該包含著測試的組織及有效資料的範圍，另外像是測試的日期及測試的項目也應該包含在內。

最後的項目就是對於執行的工作做一個結論，不論是執行的時間，得到的結果或是執行時的方法都必須包括在裏面。

10.2 系統的記錄工作

系統的記錄包含了這個系統所有的訊息，並且對於程式的設計及測試的工作也都有詳盡的描述。一個系統的說明就是系統的結構及工作的流程，這種情形與上面所描述的記錄大致相同；一個系統的流程圖(我們將在稍後討論流程圖的問題)是由許多的程式組合而成的，同時也把所有的輸入、輸出等等情況包含在內。圖 10 － 2 中就是描述系統時應注意的幾個項目：

1. 系統的摘要。
2. 系統的流程圖。
3. 程式的流程圖。
4. 輸出的訊息。
5. 執行手册。
6. 操作的指令。
7. 程式列。
8. 索引。

圖 10-2　系統的描述

當我們討論系統的時候，我們必須記得一點那就是我們實際上就是在討論一群相關的程式，它們形成了整個的系統。

10.3 程式的記錄工作

在一個程式被設計完成之後，它幾乎無法永遠的維持本來的樣子，由於機器硬體方面的改變，程式目的的改變或是考慮其他的問題，程式幾乎都要經過修改。程式的記錄工作都是由程式設計師自己完成的，供以後檢查或使用時做爲參考的工具。

一個程式的記錄包含了幾項因素；開始的第一個工作就是要指出程式的規格；一般來說就是使用一個敍述說明程式所做的工作，規格中並且包含了輸入，輸出，主程式的功能，資料的格式、範圍，以及處理錯誤的程式段等等項目。

對於程式中許多不同性質的區域，程式設計師也可以使用REM 指令說明每個區域所做的工作，以便其他人能夠在閱讀程式時一目瞭然。

10.4　幫助操作及使用者的記錄工作

著眼於幫助使用者的記錄工作一般我們稱它爲執行手册（Run Book）；執行手册中包含了像是電腦本身及週邊設備的操作方法，程式所要達成的目的，以及執行這個程式所需要使用的指令等等資料；至於實際的內容則依賴使用者本身的程度而定。

當您設計供使用者參考的記錄的時候，您必須假設這個使用者對於電腦並不是十分瞭解的；因此，在執行手册中，您必須對設置機器及使機器動作的指令詳細的說明，甚至於對 CRT 的操作也應該如對週邊設備一般的討論清楚。

雖然對一個使用者來說他（或她）並不需要對程式十分地瞭解也能夠使用它，但是在使用之前如果能夠對程式有了大致地認識，那麼使用時的效果可能更好一些，尤其對於整個程式系統而言更是如此。

在您將程式的大略情形記錄完成了之後，您必須對操作這個程式的指令詳細說明清楚；這些說明必須包括了存入程式的方法、輸入資料的型式以及在程式系統中執行不同程式的正確順序。

10.5　流程圖

一個流程圖就是使用圖形的方式表現出一連串動作的工具，這種流程圖就是由許多的說明文字及一些標準的符號所組成的。

流程圖有兩種主要的型式，一種就是程式流程圖（program flow chart），另外一種則是系統流程圖（system flow chart）。程式流程圖只是系統流程圖的一部份，它使用圖形的表示方式將電腦作業的程序顯示出來。

至於在系統流程圖中的一個單位就代表了整個程式流程圖所做的工作；在程式流程圖中我們使用一個程序說明資料處理的情形；而在系統流程圖中，我們只是指出這個程序的輸入與輸出的情形罷了！

使用流程圖說明程序的工作有優點也有缺點；優點包含了下列幾項：它並不與任何程式語言相關，您一看就能瞭解，可以消除程式或系統中許多不必要的工作。由於有以上幾個優點，因此許多人都喜歡使用它來說明程式。

至於它的缺點是因爲有些人對於圖形的表示不太能理解，因此他（或她）必須設法花費很長的時間去瞭解整個圖形的意義；同時由於流程圖的格式使得說明的圖形必須花費較大的工夫才能完成，而且它們也不能夠明確地指出那一部份是比較重要的，這些都是它的缺點。

我們並不鼓勵也不禁止流程圖的使用，我們只是希望您能夠瞭解它的功用，這可能對您有很大的幫助。

10.6 總　結

在這一章中我們告訴了您許多做記錄工作必須注意的事項，同時我們也將許多只是在大規模處理資料的系統所要使用的記錄工作也討論過了。

我們並不鼓勵您完全地使用本章中所描述的方法去做記錄的工作，我們只希望您能夠依照您自己的需要做好程式的記錄。

在圖10－1中我們說明了做記錄工作的幾個注意事項，您可以依照這些原則做好您的記錄。

我們對於不同型式的記錄也都加以分段的討論，目的就是希望您能深入的瞭解它們。

第11章

各種程式

11.0 大　綱

這一章中包含了許多的程式，供您練習及娛樂11，希望您能夠在研究它們的同時也把這些程式輸入電腦執行一下。

這些程式分爲三個主要項目；

* ＊一數學方面的
* ＊一做轉換工作的
* ＊一遊戲

這些程式我們都會給您完整的程式列及執行的例子，希望您能夠實驗它們而加以改進。

順便一提的，如果您發現了改進的方法，那麼請您通知我們，我們將非常的感謝。

11.1 數學的程式

在這一節中的數學程式都是供您使用的，它們都是提供您一種解決某種問題的方法，希望它們對您有所幫助。

11.1.1 調換行與列的位置

一個矩陣的倒置（ transpose ）就是將它的行與列互相交換位置。

假設我們有一個 4 乘 4 的矩陣，現在想要求取它的倒置，那麼我們就必須將第一列的元素放到第一行去。

最簡單的方法就是在記憶體內建立兩個同樣大小的矩陣，然後將第一個矩陣的第一列移到第二個矩陣的第一行去；這也就是圖 11 － 1 中使用的方法。

另外一種方式就是只使用一個矩陣，並且在記憶體內預留一個暫時的儲存位置供做移動這個矩陣內元素的儲放地方。

假設有下面的矩陣；

1	2	3
4	5	6
7	8	9

它的倒置是；

1	4	7
2	5	8
3	6	9

```
1   REM *** TRANSPOSE ***
2   REM *
3   REM * WRITTEN BY:JL CAMPBELL
4   REM *       DATED:05/05/78
5   REM *
6   REM * COMPUTER:APPLE II
7   REM * LANGUAGE:APPLESOFT II
8   REM *
9   REM *******************
10  TEXT : CLEAR : POKE 216,0:A =  FRE (0)
15  DIM A(25,25),B(25,25)
20  HOME : PRINT : PRINT  TAB( 8)"*** MATRIX TRANSPOSE
***": PRINT
25  PRINT "MATRIX B(R,C) = A(C,R)": PRINT
30  INPUT "ENTER ROWS IN A MATRIX = ";R: PRINT
35  INPUT "ENTER COLS IN A MATRIX = ";C: PRINT
40  VTAB (6): CALL  - 958: PRINT "** MATRIX 'A' DATA
**"
50  FOR I = 1 TO R
60  VTAB (8): CALL  - 958
70  FOR J = 1 TO C
80  PRINT "ROW ";I;" COLUMN ";J;" = ";: INPUT A(I,J)
90  NEXT J,I
95  REM * TRANSPOSE
100  VTAB (6): CALL  - 958: PRINT  TAB( 11)"*** 'B'
MATRIX ***"
110  FOR I = 1 TO R
120  FOR J = 1 TO C
130 B(J,I) = A(I,J)
140  NEXT J,I
150  REM * PRINT TRANSPOSE
160  VTAB (8): CALL  - 958:T =  INT ((40 - (R * 6)) /
2)
170  FOR I = 1 TO C
180  FOR J = 1 TO R
190  HTAB 6 * (J - 1) + T
200  PRINT B(I,J);
210  NEXT J: PRINT
220  NEXT I: PRINT
970  VTAB (20): CALL  - 958: INPUT "WANT ANOTHER
CALCULATION?(Y/N) ";A$
975  IF  LEFT$ (A$,1) = "Y" THEN 20
980  IF  LEFT$ (A$,1) = "N" THEN 990
985  GOSUB 60: GOTO 970
990  VTAB (4): CALL  - 958: VTAB (10)
995  PRINT "GOODBYE FOR NOW...": END
```

圖11-1 矩陣倒置的程式

11.1.2　矩陣的乘法

有一個較有趣的數學程式需要處理許多的數字，那就是將兩個矩陣乘在一起而得到第三個矩陣；這個程式就在圖 11 － 2 中　，它並不是很長或是很複雜的一個程式。

矩陣相乘的必備條件就是矩陣中的元素必須正確的安排；這種安排的情形如下：

1. 假設 'X' = 'A' * 'B'
2. X(r,c) = A(r,m) * B(m,c)
3. 'A' 中的行數＝'B'中的列數
4. 'X' 是一個具有 R 列及 C 行的矩陣。

除了我們所使用的方法之外尚有其他的方法可以得到兩個矩陣相乘的結果；您可以參考數值分析方面的書籍做更進一步地瞭解。

這個程式需要您輸入'A'的列數及行數，以及'B'的行數，然後就要求您輸入'A'及'B'所有的元素。乘的部份就在行號 95 到 150　的地方，至於產生的結果則顯示在螢光幕上。

```
1   REM *** MATRIX MULTIPLY ***
2   REM *
3   REM * WRITTEN BY:JL CAMPBELL
4   REM *       DATED:05/05/78
5   REM *
6   REM * COMPUTER:APPLE II
7   REM * LANGUAGE:APPLESOFT II
8   REM *
9   REM *******************
```

```
10  TEXT : CLEAR : POKE 216,0:A =  FRE (0)
15  DIM A(25,25),B(25,25),X(25,25)
20  HOME : PRINT : PRINT  TAB( 8)"*** MATRIX MULTIPLY
***": PRINT
22  PRINT "MATRIX X(R,C) = A(R,M) * B(M,C)": PRINT
24  INPUT "ENTER ROWS IN A MATRIX = ";R: PRINT
26  INPUT "ENTER COLS IN A MATRIX = ";M: PRINT
28  INPUT "ENTER COLS IN B MATRIX = ";C: PRINT
30  VTAB (6): CALL  - 958: PRINT "** MATRIX 'A' DATA
**"
40  FOR I = 1 TO R
45  VTAB (8): CALL  - 958
50  FOR J = 1 TO M
55  PRINT "ROW ";I;" COLUMN ";J;" = ";: INPUT A(I,J)
60  NEXT J,I
65  VTAB (6): CALL  - 958: PRINT "** MATRIX 'B' DATA
**"
70  FOR I = 1 TO M
75  VTAB (8): CALL  - 958
80  FOR J = 1 TO C
85  PRINT "ROW ";I;" COLUMN ";J;" = ";: INPUT B(I,J)
90  NEXT J,I
95  REM * MULTIPLY A*B
100  VTAB (6): CALL  - 958: PRINT  TAB( 11)"*** 'X'
MATRIX ***"
110  FOR I = 1 TO R
120  FOR J = 1 TO C
125 X(I,J) = 0
130  FOR K = 1 TO M
140 X(I,J) = X(I,J) + A(I,K) * B(K,J)
150  NEXT K,J,I
160  VTAB (8): CALL  - 958:T =  INT ((40 - (C * 6)) /
2)
170  FOR I = 1 TO R
180  FOR J = 1 TO C
190  HTAB 6 * (J - 1) + T
200  PRINT X(I,J);
210  NEXT J: PRINT
220  NEXT I: PRINT
970  VTAB (20): CALL  - 958: INPUT "WANT ANOTHER
CALCULATION?(Y/N) ";A$
975  IF  LEFT$ (A$,1) = "Y" THEN 20
980  IF  LEFT$ (A$,1) = "N" THEN 990
985  GOSUB 60: GOTO 970
990  VTAB (4): CALL  - 958: VTAB (10)
995  PRINT "GOODBYE FOR NOW...": END
```

圖11-2　矩陣相乘的程式

11.1.3　NEWTON-RAPHSON方法

Newton-Raphson 方法是使用反覆計算的方式求得一個方程式實根的逼近值。下面是使用的公式：

F(n+1) = X(n) - F(X(n)) / F'(X(n))

F(X)就是函數，F'(X) 是這個函數的第一次微分，n則是反覆計算的次數。

使用在第一次計算中得到的結果做爲第二次計算中的函數值，您就可以反覆地使用這種方式求得在允許範圍內誤差值（由您決定）的實根了。

```
1   REM *** NEWTON RAPHSON ***
2   REM *
3   REM * XL=LEFT BOUND
4   REM *
5   REM * DEF FCN IN #10
6   REM * DEF F'(FCN) IN #20
7   REM * AER=ACCEPTABLE ERROR
8   REM *
9   REM **********************
10   DEF   FN A(X) = X ^ 3 - 75
20   DEF   FN F(X) = 3 * X ^ 2
25   HOME : PRINT : PRINT  TAB( 9)"*** NEWTON RAPHSON
***": PRINT
30   INPUT "ENTER LEFT  BOUND=XL  ";XL
32   INPUT "ENTER ERROR BOUND=AER ";AER
34   PRINT : PRINT
40   PRINT "N     F1      XL      OMEGA"
45   PRINT
50   FOR  N = 0  TO  150
60  FD =  FN F(XL)
70   IF  FD = 0  THEN  150
80  F1 =  FN A(XL)
90  OMEGA = F1 / FD
100   IF  ABS (OMEGA) < AER THEN 170
110   PRINT N;" ";F1;" ";XL;" ";OMEGA
120  XL = XL - OMEGA
130   NEXT  N
140   GOTO  170
150   PRINT  "FD=0"
170   END
```

圖11-13　Newton-Raphson 程式

您在程式的 10 及 20 的行號處設定函數及它的第 1 次微分，接著在執行的時候您就被要求輸入左邊界限的起始點 XL，及可以被容許的誤差值，AER 。

然後這個程式就會最多反覆地執行 150 次這個公式，並且在每執行一次之前更換 XL 的值，直到逼近到容許的範圍內爲止。

11.1.4　搜尋方法 (SEARCH METHOD)

搜尋方法就如同它的名字一般，在一個搜尋的範圍內找出一個多項式所有的根。

您在程式中行號 10 的地方定義這個多項式；並且在程式被執行的時候輸入左邊及右邊的搜尋界限，在這兩個界限內您就可以找到這個多項式在這個範圍內的根了；接著您就必須輸入設定這個範圍內小區間數目的值。

程序：

假設一個多項式，f(X)，在 (XR,XL) 的範圍內是連續的，並且 f(X) * f(XR) <=0；那麼下面就是求根的程序：

1. 設定 F1 = FN F(XL)
2. X = XL,XL+ 　　增加數，XL+2*增加數，…… UNTIL 滿意爲止。
3. 計算 F2 = FN F(X + 增加數）。
4. 如果 F1 * F2 > 0 則第 6 步。
5. 印出計算值及函數值。
6. F1 = F2
7. NEXT X

只要您找到了一個根，您就可以將範圍縮小，增加小區間的數目，再重新執行這個程式；使用這種方法您就可以得到較精確的根了。

這種方法執行的速度並不如 **Newton-Raphson** 等方法來得快；但是，它卻給了您一個在極小區間內求根的方法，這使得您在使用其他的方法解決問題的時候會覺得更快速，結果也許更好一些。

```
1   REM *** SEARCH METHOD ***
2   REM *
3   REM * XL=LEFT BOUND
4   REM * XR=RIGHT BOUND
5   REM *  N= # OF SUBINTERVALS
6   REM * DEFINE FCN IN STMT #10
7   REM * WRITTEN BY:JL CAMPBELL
8   REM *       DATED:02/03/78
9   REM *********************
10  DEF   FN F(X) = X ^ 3 - X - 1
20  HOME
25  PRINT : PRINT  TAB( 10)"**** SEARCH METHOD ****":
PRINT
30  PRINT : INPUT "ENTER LEFT  BOUND=XL  ";XL
31  INPUT "ENTER RIGHT BOUND=XR  ";XR
32  INPUT "ENTER SUBINTERVALS=N  ";N: PRINT
35  PRINT "CNT";: PRINT  TAB( 5)"X";: PRINT  TAB(
17)"NS";: PRINT  TAB( 29)"FNF(X+NS/2)"
40 F1 =  FN F(XL)
50 CNT = 1
60 NS = (XR - XL) / N
70  FOR  X = XL  TO  (XR - NS)  STEP  NS
80 F2 =  FN F(X + NS)
90  IF  F1 * F2 > 0  THEN  120
100  PRINT CNT;: PRINT  TAB( 4)X;: PRINT  TAB( 16)X +
NS;: PRINT  TAB( 28) FN F(X + NS / 2)
110 CNT = CNT + 1
120 F1 = F2
130  NEXT  X
140  STOP
150  END
```

圖11-4 搜尋的程式

11.1.5 BISECTION方法

這個方法也是用來找到一個多項式的根，但是它的速度較前面所描述的搜尋的方法要快得多了。

程序：

假設一個函數， f(x) ，在 (XL,XR) 的範圍內是連續的，並且 f(XL) * f(XR) <= 0。下面就是求這個函數根的程序；

1. 設定 FN a(x) ＝函數。
2. FOR n = 0,1,2,3....UNTIL　滿意爲止，執行
3. 設定 m = (XL + XR)/2
4. IF f(XLn * f(m) <= 0 THEN XLn+1 = XLn: XRn+1 = m: GOTO 6
5. ELSE XLn+1 = m: XRn+1 = XRn
6. f(x) 在 (XLn+1,XRn+1) 的範圍內有一個根。

圖11-5　Bisection 方法

每反覆地執行一次的時候，包含著根的範圍就被縮小一半，這樣子使得我們逼近根的速度變得很快。

請注意這個函數在行號 10 的地方先被定義；當程式被執行的時候，您將被詢問左、右邊的界限，以及可以被接受的誤差值；這個誤差值的意義是說您可以由完全精確的情況到一個可以接受的絕對誤差。例如，您輸入的 ER = 0.1 ，這將使得任何一個求得的 X 值可以具有正負 0.1 的不準確值。

因此，如果 f(X) = 0.08 ，您就已經找到了一個具有可以被接受誤差值的根了。當您改變 E R 的值，您也就是要求程式爲您找出適合您指定的誤差的根了。

注意：我們在第七章的 7.7.2 節中曾經討論過了二分法搜尋的方式；目前我們所使用的這個求根的方法，原則上與它相同。

```
1   REM *** BISECTION METHOD ***
2   REM *
3   REM * XL=LEFT BOUND
4   REM * XR=RIGHT BOUND
5   REM *
6   REM * DEFINE FCN IN #10
7   REM * WRITTEN BY:JL CAMPBELL
8   REM *       DATED:02/05/78
9   REM ************************
10  DEF  FN A(X) = X ^ 3 - 75
20  HOME : PRINT : PRINT  TAB( 8)"*** BISECTION METHOD
***": PRINT
25  INPUT "ENTER  LEFT BOUND=XL ";XL
28  INPUT "ENTER RIGHT BOUND=XR ";XR
30  INPUT "ENTER ERROR FCN.=ER  ";ER: PRINT
40 X1 = XL
50 X2 = XR
60 F1 =  FN A(X1)
70 F2 =  FN A(X2)
75  PRINT "N";: PRINT  TAB( 5)"M";: PRINT  TAB( 20)"FN
A(M)": PRINT
80  IF  F1 * F2 < 0  THEN  110
90  PRINT  "F1*F2 IS NOT NEGATIVE"
100  GOTO  260
110  FOR  N = 0  TO  40
120 M = (X1 + X2) / 2
130 F =  FN A(M)
140  IF  ABS (F) < ER  THEN  260
170  PRINT N;: PRINT  TAB( 4)M;: PRINT  TAB( 19)F
180  IF  F1 * F > 0  THEN  220
190 F2 = F
200 X2 = M
210  GOTO  240
220 F1 = F
230 X1 = M
240  NEXT  N
260  END
```

圖11-6　Bisection 求根的程式

11.1.6 FIBONACCI數列

FIBONACCI　數列是一個無窮數列，它就像下面的型式一般；

0,1,1,2,3,5,8,13,21,34.....

這個數列的起始點就是零，第二個項目則是1，其後所有的項目都是依照下面的公式得來的；

$$X(n) = X(n-1) + X(n-2)$$

因此，這個數列的第10項就是；

$$X(10) = X(9) + X(8) = 21 + 13 = 34$$

圖11－7　就是求得這個數列的程式，我們只是找出前面的26 項，這是因爲X 這個向量在 DIM 敍述中只被定爲25的原因。

計算的步驟是在行號80到110的部份完成的。

產生的結果則在行號120到180的地方顯示在螢光幕上。

結果；

```
          *** FIBONACCI SERIES ***

 THIS PROGRAM WILL FIND THE FIRST 26
 TERMS IN A FIBONACCI SERIES.

   0
   1      1      2      3      5
   8      13     21     34     55
   89     144    233    377    610
   987    1597   2584   4181   6765
   10946  17711  28657  46368  75025
```

```
1   REM *** FIBONACCI SERIES ***
2   REM *
3   REM * WRITTEN BY:JL CAMPBELL
4   REM *        DATED:07/21/80
5   REM *
6   REM * COMPUTER:APPLE II
7   REM * LANGUAGE:APPLESOFT II
8   REM *
9   REM ********************
10   TEXT : CLEAR : POKE 216,0:A =  FRE (0)
20   DIM X(25)
50   HOME : PRINT : PRINT  TAB( 8)"*** FIBONACCI SERIES
***": PRINT
```

```
60  PRINT "THIS PROGRAAM WILL FIND THE FIRST 26"
70  PRINT "TERMS IN A FIBONACCI SERIES."
80 X(0) = 0:X(1) = 1
90  FOR I = 2 TO 25
100 X(I) = X(I - 1) + X(I - 2)
110  NEXT I
120  VTAB (7): CALL  - 958
125  HTAB 3: PRINT X(0)
130  FOR I = 1 TO 5
140 T = 3
150  FOR J = (I - 1) * 5 + 1 TO (I - 1) * 5 + 5
160  HTAB T: PRINT X(J);
165 T = T + 6
170  NEXT J: PRINT
180  NEXT I
190  END
```

圖11-7 Fibonnacci數列

11.1.7 高斯消去法(Gauss Elim ination)

在這一節中我們將討論解聯立方程式的方法，雖然這種方法比較古老，但是它仍然是最有效率、最直接的方法之一。

考慮下面的聯立方程式：

```
+-                  -+ +-   -+    +-   -+
! C11 C12 C13 C14 ! ! X1 !    ! R1 !
! C21 C22 C23 C24 ! ! X2 !    ! R2 !
! C31 C32 C33 C34 ! ! X3 ! = ! R3 !
! C41 C42 C43 C44 ! ! X4 !    ! R4 !
+-                  -+ +-   -+    +-   -+
```

我們研究一下矩陣的特性，您可以知道這個存放根的向量，X，雖然您對它做了以下的動作其值仍然不變：

1. 對於任何一個方程式乘上或除以一個常數。
2. 使用一個方程式與其他方程式的和或差代替這個方程式。

藉著對每一列重複地作業，我們就可以得到這一個聯立方程式的解了首先我們使用C11除第1列。

```
+-                            -+ +-   -+    +-    -+
! 1    C'12 C'13 C'14 ! ! X1 !    ! R'1 !
! C21 C22  C23  C24  ! ! X2 !    ! R2  !
! C31 C32  C33  C34  ! ! X3 ! =  ! R3  !
! C41 C42  C43  C44  ! ! X4 !    ! R4  !
+-                            -+ +-   -+    +-    -+
```

現在我們就可以將C21乘上第一列所有的值然後減去第二列相對應的值了。

```
+-                            -+ +-   -+    +-    -+
! 1    C'12 C'13 C'14 ! ! X1 !    ! R'1 !
! 0    C'22 C'23 C'24 ! ! X2 !    ! R'2 !
! C31 C32  C33  C34  ! ! X3 ! =  ! R3  !
! C41 C42  C43  C44  ! ! X4 !    ! R4  !
+-                            -+ +-   -+    +-    -+
```

我們現在可以對第3及第4個方程式做同樣地的動作；我們將第1個方程式乘上 C31 ，再減去第3個方程式；然後再將 C41 乘上第1個方程式，再減去第4個方程式。第1個方程式就是中心列（ pivot row ），而元素 C11 就是第1個方程式的中心元素（ pivot element ）。經過上面作業產生的結果就如：

```
+-                        -+ +-   -+    +-    -+
! 1 C'12 C'13 C'14 ! ! X1 !    ! R'1 !
! 0 C'22 C'23 C'24 ! ! X2 !    ! R'2 !
! 0 C'32 C'33 C'34 ! ! X3 ! =  ! R'3 !
! 0 C'42 C'43 C'44 ! ! X4 !    ! R'4 !
+-                        -+ +-   -+    +-    -+
```

現在我們可以再往第2列作業了；在第二列中元素C'22就是它的中心元素，然後利用上面所描述的方法做同樣減的動作。當第2列的作業完成了之後，我們就往第3及第4列繼續執行；最後我們得到下面的結果：

```
+-                          -+ +-   -+    +-    -+
! 1    C'12 C'13 C'14 ! ! X1 !    ! R'1 !
!      1    C'23 C'24 ! ! X2 !    ! R'2 !
!           1    C'34 ! ! X3 !  = ! R'3 !
!                1    ! ! X4 !    ! R'4 !
+-                          -+ +-   -+    +-    -+
```

我們由上面矩陣中的最後一個式子知道 X4 可以直接得到，它就是：

X4 = R'4

X3 則可以使用下面的計算得到：

X3 = R'3 - C'34*X4

得到 X3 與 X4 之後，我們可以求得 X2 。

X2 = R'2 - C'24*X4 - C'23*X3

最後，我們計算 X1 。

X1 = R'1 - C'14*X4 - C'13*X3 - C'12*X2

```
1   REM   * GAUSS ELIMINATION FOR
2   REM   * N EQNS. IN N UNKNOWNS.
3   REM   * "NUMERICAL METHODS" BY
4   REM   * ROBERT W. HORNBECK PG 94
5   REM   * ENTER COEFF. IN STMT 50 AS ASKED
6   REM   * ENTER # OF EQNS AS PROMPTED
7   REM   * ENTER R(I) AS REQUIRED
8   REM   *
9   HOME
10   INPUT  "ENTER # OF EQNS.";N
20   DIM  C(N,N),X(N),R(N)
30   FOR  I = 1  TO  N
40   FOR  J = 1  TO  N
50   PRINT  "ENTER C(";I;",";J;")=";
55   INPUT  C(I,J)
60   NEXT  J,I
70   FOR  I = 1  TO  N
80   PRINT  "ENTER R(";I;")=";
85   INPUT  R(I)
90   NEXT  I
```

```
100 K = 1
110 TEMP = 1.0 / C(K,K)
120 J = K
130 C(K,J) = C(K,J) * TEMP
135  IF   ABS (C(K,J))  < 1.0E - 05  THEN  C(K,J) = 0
140  IF  J = N  THEN  170
150 J = J + 1
160  GOTO  130
170 R(K) = R(K) * TEMP
180 J = K + 1
190 TEMP = C(J,K)
200 L = K
210 C(J,L) = C(J,L) - C(K,L) * TEMP
215  IF   ABS (C(J,L)) < 1.0E - 05  THEN  C(J,L) = 0
220  IF  L = N  THEN  250
230 L = L + 1
240  GOTO  210
250 R(J) = R(J) - R(K) * TEMP
260  IF  J = N  THEN  290
270 J = J + 1
280  GOTO  190
290  IF  K = N - 1  THEN  320
300 K = K + 1
310  GOTO  110
320 X(N) = R(N) / C(N,N)
330 I = 1
340 SUM = 0
350 J = N - I + 1
360 SUM = SUM + C(N - I,J) * X(J)
370  IF  J = N  THEN  400
380 J = J + 1
390  GOTO  360
400 X(N - I) = R(N - I) - SUM
410  IF  I = N - 1  THEN  440
420 I = I + 1
430  GOTO  340
440  HOME
450  PRINT "    FINAL ARRAY": PRINT
460  FOR  I = 1  TO  N
470  FOR  J = 1  TO  N
480  PRINT  TAB( 1 + (J - 1) * 12)C(I,J);
490  NEXT  J: PRINT : NEXT  I: PRINT
500  PRINT : PRINT  "X(I) SOLNS.": PRINT
510  FOR  I = 1  TO  N
520  PRINT  "X(";I;")=";X(I)
530  NEXT  I
540  STOP
550  END
```

圖11-18　高斯消去法程式

11.2 轉換的程式

我們在這一節中將提供您幾個轉換的程式，它們都是由一種計量的方式轉換成另外一種計量方式的程式。

由於我們都是由 USA 制轉換到公制，或是由華氏轉換到攝氏，相信它們對您來說應該是相當有趣的。

11.2.1 溫 度

溫度的轉換是非常直接的，我們使用的是下面的公式；

C = 攝氏溫度

F = 華氏溫度

$$C = (F - 32) * 5/9$$

$$F = (C * (9/5)) + 32$$

華氏 32 度也就是攝氏的零度。

```
 1  REM *** TEMPERATURES ***
 2  REM *
 3  REM * WRITTEN BY:JL CAMPBELL
 4  REM *       DATED:05/24/78
 5  REM *
 6  REM * COMPUTER:APPLE II
 7  REM * LANGUAGE:APPLESOFT II
 8  REM *
 9  REM *******************
10  TEXT : CLEAR : POKE 216,0:A =  FRE (0)
50  GOTO 900
51  REM **************
52  REM * GEN'L SUBR *
53  REM **************
55  INPUT "WHICH ONE? ==> ";S: RETURN
60  VTAB (22): HTAB 10: INVERSE : PRINT "ILLEGAL
ENTRY": NORMAL : PRINT  CHR$ (7)
65  FOR L = 1 TO 1000: NEXT L: RETURN
```

```
899  REM * MAIN DRIVER
900  HOME : PRINT : PRINT  TAB( 10)"*** TEMPERATURES
***": PRINT
910  PRINT "MENU:": PRINT "----"
912  PRINT  TAB( 5)"1- TO CELSIUS": PRINT
914  PRINT  TAB( 5)"2- TO FAHRENHEIT": PRINT
920  PRINT  TAB( 5)"0- EXIT SYSTEM"
922  VTAB (12): CALL  - 958: GOSUB 55
924  IF S <  > 0 THEN 930
926  VTAB (4): CALL  - 958: VTAB (10)
928  PRINT "GOODBYE FOR NOW...": END
930  IF S > 0 AND S < 3 THEN 940
932  GOSUB 60: GOTO 922
940  ON S GOSUB 1000,2000
970  VTAB (20): CALL  - 958: INPUT "WANT ANOTHER
CALCULATION?(Y/N) ";A$
975  IF  LEFT$ (A$,1) = "Y" THEN 900
980  IF  LEFT$ (A$,1) = "N" THEN 926
985  GOSUB 60: GOTO 970
1000  HOME : PRINT : PRINT  TAB( 7)"*** CONVERT TO
CELSIUS ***": PRINT
1010  INPUT "ENTER FAHRENHEIT DEGREES = ";F
1020 C = (F - 32) * (5 / 9)
1030 C =  INT (C * 100 + 0.5) / 100
1040  PRINT : PRINT "CELSIUS DEGREES = ";C
1995  RETURN
2000  HOME : PRINT : PRINT  TAB( 5)"*** CONVERT TO
FAHRENHEIT ***": PRINT
2010  INPUT "ENTER CELSIUS DEGREES  = ";C
2020 F = (C * (9 / 5)) + 32
2030 F =  INT (F * 100 + 0.5) / 100
2040  PRINT : PRINT "FAHRENHEIT DEGREES = ";F
2995  RETURN
```

圖11-9　溫度轉換成的程式

11.2.2　度度量轉換成弳度量 (ANGLES TO RADIANS)

這個程式將度度量轉換成弳度量，您必須輸入度（ degrees ），分（ minutes）及秒（ seconds ），而程式將它轉換成弳度量。

轉換的工作是非常直接的，就在程式中行號 70 到130 的地方完成。

```
1  REM *** RADIANS TO ANGLE ***
2  REM *
3  REM * WRITTEN BY:JL CAMPBELL
4  REM *       DATED:07/21/80
5  REM *
6  REM * COMPUTER:APPLE II
7  REM * LANGUAGE:APPLESOFT II
8  REM *
9  REM ******************
10  TEXT : CLEAR : POKE 216,0:A =  FRE (0)
20 PI = 3.1415927
50  HOME : PRINT : PRINT  TAB( 7)"*** RADIANS TO
DEGREES ***": PRINT
60  INPUT "ENTER RADIANS = ";R: PRINT
70 A = 3600 * 180 * R / PI
80 D1 =  INT (A / 3600)
90 C =  INT (D1 / 360)
100 D = D1 - 360 * C
110 M =  INT ((A - D1 * 3600) / 60)
120 S = A - D1 * 3600 - ( INT ((A - D1 * 3600) / 60)) *
60
130 S =  INT (S * 100 + 0.5) / 100
140  PRINT "DEGREES = ";D
150  PRINT "MINUTES = ";M
160  PRINT "SECONDS = ";S
170  VTAB (10): INPUT "ANOTHER PROBLEM?(Y/N) ";A$
180  IF  LEFT$ (A$,1) = "Y" THEN 50
190  VTAB (12): PRINT "GOODBYE FOR NOW...": END
```

圖11-10 度度量轉換成弳度量

11.2.3 弳度量轉換成度度量(RANDIANS TO ANGLES)

這個程式與上面的程式剛好做相反的作業。

您輸入弳度量R，程式將給您度度量的值。

```
1  REM *** ANGLE TO RADIANS ***
2  REM *
3  REM * WRITTEN BY:JL CAMPBELL
4  REM *       DATED:07/21/80
5  REM *
6  REM * COMPUTER:APPLE II
7  REM * LANGUAGE:APPLESOFT II
8  REM *
9  REM *******************
10  TEXT : CLEAR : POKE 216,0:A =  FRE (0)
20 PI = 3.1415926
50  HOME : PRINT : PRINT  TAB( 7)"*** DEGREES TO
RADIANS ***": PRINT
60  INPUT "ENTER DEGREES,MINUTES,SECONDS = ";D,M,S
70 A = D + M / 60 + S / 3600
80 C =  INT (A / 360)
90  VTAB (6): PRINT "RADIANS = ";A * (2 * PI / 360)
100  VTAB (8): INPUT "ANOTHER PROBLEM?(Y/N) ";A$
110  IF  LEFT$ (A$,1) = "Y" THEN 50
120  VTAB (10): PRINT "GOODBYE FOR NOW...": END
```

圖11-11 弳度量轉換成度度量

11.2.4 公 制

這個程式將英制的數量轉換成公制的數量，同時您也可以做相反的動作；轉換的資料都放在行號 812 到 844 的 DATA 敍述內。這個程式的整個組織情形就如同圖 4 － 8 中的架構程式一般。

手册、計算、以及一行的副程式都放在一般的副程式區域內；至於主程式則位於行號 900 到 990 的地方。

首先您必須對您所要做的轉換工作做一個選擇，接著才是選擇公制或英制的轉換；當您做完了這兩個選擇之後，程式就依照轉換表而將您的要求完成了。

```
1   REM ***** CONVERSIONS *****
2   REM *
3   REM * WRITTEN BY:JL CAMPBELL
4   REM *        DATED:08/10/81
5   REM *
6   REM * LANGUAGE:APPLESOFT II
7   REM * COMPUTER:APPLE II
8   REM *
9   REM ***********************
10  TEXT : CLEAR : POKE 216,0:A =  FRE (0)
20  DIM L$(5,3),L(5,3),V$(2,3),V(2,3),
     M$(5,3),M(5,3),A$(3,3),A(3,3)
50  GOTO 800
51  REM **************
52  REM * GEN'L SUBR *
53  REM **************
55  INPUT "WHICH ONE? = ";N:N =  INT (N): RETURN
60  VTAB 22: CALL  - 958: HTAB 10: INVERSE : PRINT
"ILLEGAL ENTRY!"; CHR$ (7): NORMAL
64  FOR D = 1 TO 1000: NEXT D: RETURN
70  VTAB 4: CALL  - 958: PRINT "CONVERT:": PRINT
"-------": RETURN
75  VTAB 14: CALL  - 958: PRINT "ENTER NUMBER OF
";L$(N,P);: RETURN
80  VTAB 14: CALL  - 958: PRINT "ENTER NUMBER OF
";V$(N,P);: RETURN
85  VTAB 14: CALL  - 958: PRINT "ENTER NUMBER OF
";M$(N,P);"S";: RETURN
90  VTAB 14: CALL  - 958: PRINT "ENTER NUMBER OF
";A$(N,P);: RETURN
99  REM * CALC QUESTION SUBR
100 SW = 0
105  VTAB 20: CALL  - 958: INPUT "ANOTHER
     CALCULATION?(Y/N) ";Q$
110  IF  LEFT$ (Q$,1) = "Y" THEN SW = 1: GOTO 125
115  IF  LEFT$ (Q$,1) = "N" THEN SW = 0: GOTO 125
120  GOSUB 60: GOTO 105
125  RETURN
199  REM * SUB MENU SUBR
200  PRINT "SUB MENU:": PRINT "--- ----"
205  PRINT  TAB( 9)"1- CONVERT TO ANGLO": PRINT
210  PRINT  TAB( 9)"2- CONVERT TO METRIC": PRINT
215  VTAB 10: PRINT  TAB( 9)"0- EXIT SECTION"
220  RETURN
299  REM * CONV SUBR
300 Y = L(N,1) * L(N,3) * X:Z = L(N,1) * L(N,2) * X
```

```
305  RETURN
310 Y = V(N,1) * V(N,3) * X:Z = V(N,1) * V(N,2) * X
315  RETURN
320 Y = M(N,1) * M(N,3) * X:Z = M(N,1) * M(N,2) * X
325  RETURN
330 Y = A(N,1) * A(N,3) * X:Z = A(N,1) * A(N,2) * X
335  RETURN
340 Y = (L(N,1) / L(N,3)) * X:Z = (L(N,2) / L(N,3)) * X
345  RETURN
350 Y = (V(N,1) / V(N,3)) * X:Z = (V(N,2) / V(N,3)) * X
355  RETURN
360 Y = (M(N,1) / M(N,3)) * X:Z = (M(N,2) / M(N,3)) * X
365  RETURN
370 Y = (A(N,1) / A(N,3)) * X:Z = (A(N,2) / A(N,3)) * X
375  RETURN
799  REM * READ DATA
800  HOME : VTAB 12: HTAB 13: INVERSE :
      PRINT "LOADING TABLES": NORMAL
810  FOR I = 1 TO 5: FOR J = 1 TO 3: READ L$(I,J): READ
      L(I,J): NEXT J,I
812  DATA  MIL,1,INCH,0.001,CM,0.00254
813  DATA  INCH,1,MIL,1000,CM,2.54
814  DATA  FEET,1,INCHES,12,CM,30.48
815  DATA  YARD,1,FEET,3,M,0.9144
816  DATA  MILE,1,YARDS,1760,KM,1.6093
820  FOR I = 1 TO 2: FOR J = 1 TO 3: READ V$(I,J): READ
      V(I,J): NEXT J,I
822  DATA  IN^3,1,CM^3,16.39,L^3,0.01639
823  DATA  FT^3,1,IN^3,1728,L^3,28.32
830  FOR I = 1 TO 5: FOR J = 1 TO 3: READ M$(I,J): READ
      M(I,J): NEXT J,I
832  DATA  GRAIN,1,GRAM,0.0648,OZ,0.00228551
833  DATA  DRAM,1,GRAM,1.772,OZ,0.06249844
834  DATA  OZ,1,DRAM,16,GRAM,28.35
835  DATA  LB,1,OZ,16,GRAM,453.6
836  DATA  TON,1,LB,2000,KG,1016.1
840  FOR I = 1 TO 3: FOR J = 1 TO 3: READ A$(I,J): READ
      A(I,J): NEXT J,I
842  DATA  IN^2,1,CM^2,6.452,M^2,0.006452
843  DATA  FT^2,1,IN^2,144,CM^2,0.0929
844  DATA  YD^2,1,FT^2,9,M^2,0.8361
899  REM * MAIN DRIVER
900  HOME : PRINT : PRINT  TAB( 10)"*** CONVERSIONS
      ***": PRINT
905  PRINT "MAIN MENU:": PRINT "---- ----"
910  PRINT  TAB( 10)"1- LENGTH": PRINT
```

```
915  PRINT  TAB( 10)"2- VOLUME": PRINT
920  PRINT  TAB( 10)"3- MASS": PRINT
925  PRINT  TAB( 10)"4- AREA": PRINT
930  PRINT  TAB( 10)"0- EXIT PROGRAM"
935  VTAB 16: CALL  - 958: GOSUB 55
940  IF N = 0 THEN 990
945  IF N > 0 AND N < 5 THEN 955
950  GOSUB 60: GOTO 935
955  ON N GOSUB 1000,2000,3000,4000
960  GOTO 900
990  VTAB 4: CALL  - 958: VTAB 12: PRINT
"SEE'YA..........HONEY!": END
1000  HOME : PRINT : PRINT  TAB( 12)"*** LENGTH
       ***": PRINT
1010  GOSUB 200
1025  VTAB 12: CALL  - 958: GOSUB 55
1030  IF N = 0 THEN  RETURN
1035  IF N > 0 AND N < 3 THEN 1050
1040  GOSUB 60: GOTO 1025
1050  ON N GOSUB 1100,1600
1060  GOTO 1000
1100  GOSUB 70:P = 3
1105  FOR I = 1 TO 5: HTAB 8
1110  PRINT I;"- ";L$(I,3);: HTAB 14: PRINT "==>";:
       HTAB 18: PRINT L$(I,1)
1115  NEXT I
1120  VTAB 12: HTAB 8: PRINT "0- EXIT SECTION"
1125  VTAB 14: CALL  - 958: GOSUB 55
1130  IF N = 0 THEN 1195
1135  IF N > 0 AND N < 6 THEN 1145
1140  GOSUB 60: GOTO 1125
1145  ON N GOSUB 1200,1250,1300,1350,1350
1150  GOSUB 100
1160  IF SW = 1 THEN 1125
1195  RETURN
1200  GOSUB 75: INPUT "S = ";X
1210  GOSUB 340
1220  VTAB 16: PRINT X;" ";L$(N,3);"S = ";Y;" ";L$(N,1)
1240  VTAB 18: PRINT X;" ";L$(N,3);"S = ";
       Z;" ";L$(N,2);"ES"
1245  RETURN
1250  GOSUB 75: INPUT "S = ";X
1260  GOSUB 340
1270  VTAB 16: PRINT X;" ";L$(N,3);"S = ";
       Y;" ";L$(N,1);"ES"
1290  VTAB 18: PRINT X;" ";L$(N,3);"S = ";
       Z;" ";L$(N,2);"S"
```

```
1295  RETURN
1300  GOSUB 75: INPUT "S = ";X
1310  GOSUB 340
1320  VTAB 16: PRINT X;" ";L$(N,3);"S = ";Y;" ";L$(N,1)
1340  VTAB 18: PRINT X;" ";L$(N,3);"S = ";Z;" ";L$(N,2)
1345  RETURN
1350  GOSUB 75: INPUT "S = ";X
1360  GOSUB 340
1370  VTAB 16: PRINT X;" ";L$(N,3);"S = ";
       Y;" ";L$(N,1);"S"
1390  VTAB 18: PRINT X;" ";L$(N,3);"S = ";Z;" ";L$(N,2)
1395  RETURN
1600  GOSUB 70:P = 1
1605  FOR I = 1 TO 5: HTAB 8
1610  PRINT I;"- ";L$(I,1);: HTAB 16: PRINT "==>";:
       HTAB 20: PRINT L$(I,3)
1615  NEXT I
1620  VTAB 12: HTAB 8: PRINT "0- EXIT SECTION"
1625  VTAB 14: CALL  - 958: GOSUB 55
1630  IF N = 0 THEN  RETURN
1635  IF N > 0 AND N < 6 THEN 1645
1640  GOSUB 60: GOTO 1625
1645  ON N GOSUB 1700,1750,1800,1850,1850
1650  GOSUB 100
1660  IF SW = 1 THEN 1625
1695  RETURN
1700  GOSUB 75: INPUT "S = ";X
1710  GOSUB 300
1720  VTAB 16: PRINT X;" ";L$(N,1);"S = ";Y;" ";L$(N,3)
1740  VTAB 18: PRINT X;" ";L$(N,1);"S = ";
       Z;" ";L$(N,2);"ES"
1745  RETURN
1750  GOSUB 75: INPUT "ES = ";X
1760  GOSUB 300
1770  VTAB 16: PRINT X;" ";L$(N,1);"ES = ";
       Y;" ";L$(N,3)
1790  VTAB 18: PRINT X;" ";L$(N,1);"ES = ";
       Z;" ";L$(N,2);"S"
1795  RETURN
1800  GOSUB 75: INPUT " = ";X
1810  GOSUB 300
1820  VTAB 16: PRINT X;" ";L$(N,1);" = ";Y;" ";L$(N,3)
1840  VTAB 18: PRINT X;" ";L$(N,1);" = ";Z;" ";L$(N,2)
1845  RETURN
1850  GOSUB 75: INPUT "S = ";X
1860  GOSUB 300
```

```
1870  VTAB 16: PRINT X;" ";L$(N,1);"S = ";
       Y;" ";L$(N,3);"ETERS"
1890  VTAB 18: PRINT X;" ";L$(N,1);"S = ";Z;" ";L$(N,2
1895  RETURN
2000  HOME : PRINT : PRINT  TAB( 13)"*** VOLUME
       ***": PRINT
2010  GOSUB 200
2025  VTAB 12: CALL  - 958: GOSUB 55
2030  IF N = 0 THEN  RETURN
2035  IF N > 0 AND N < 3 THEN 2050
2040  GOSUB 60: GOTO 2025
2050  ON N GOSUB 2100,2600
2060  GOTO 2000
2100  GOSUB 70:P = 3
2105  FOR I = 1 TO 2: HTAB 8
2110  PRINT I;"- ";V$(I,3);: HTAB 15: PRINT "==>";:
       HTAB 19: PRINT V$(I,1)
2115  NEXT I
2120  VTAB 12: HTAB 8: PRINT "0- EXIT SECTION"
2125  VTAB 14: CALL  - 958: GOSUB 55
2130  IF N = 0 THEN  RETURN
2135  IF N > 0 AND N < 3 THEN 2145
2140  GOSUB 60: GOTO 2125
2145  GOSUB 2200
2150  GOSUB 100
2160  IF SW = 1 THEN 2125
2195  RETURN
2200  GOSUB 80: INPUT " = ";X
2210  GOSUB 350
2220  VTAB 16: PRINT X;" ";V$(N,3);" = ";Y;" ";V$(N,1)
2240  VTAB 18: PRINT X;" ";V$(N,3);" = ";Z;" ";V$(N,2)
2245  RETURN
2600  GOSUB 70:P = 1
2605  FOR I = 1 TO 2: HTAB 8
2610  PRINT I;"- ";V$(I,1);: HTAB 16: PRINT "==>";:
       HTAB 20: PRINT V$(I,3)
2615  NEXT I
2620  VTAB 12: HTAB 8: PRINT "0- EXIT SECTION"
2625  VTAB 14: CALL  - 958: GOSUB 55
2630  IF N = 0 THEN  RETURN
2635  IF N > 0 AND N < 3 THEN 2645
2640  GOSUB 60: GOTO 2625
2645  GOSUB 2700
2650  GOSUB 100
2660  IF SW = 1 THEN 2625
2695  RETURN
2700  GOSUB 80: INPUT " = ";X
```

```
2710  GOSUB 350
2720  VTAB 16: PRINT X;" ";V$(N,1);" = ";Y;" ";V$(N,3)
2740  VTAB 18: PRINT X;" ";V$(N,1);" = ";Z;" ";V$(N,2)
2745  RETURN
3000  HOME : PRINT : PRINT  TAB( 14)"*** MASS
       ***": PRINT
3010  GOSUB 200
3025  VTAB 12: CALL  - 958: GOSUB 55
3030  IF N = 0 THEN  RETURN
3035  IF N > 0 AND N < 3 THEN 3050
3040  GOSUB 60: GOTO 3025
3050  ON N GOSUB 3100,3600
3060  GOTO 3000
3100  GOSUB 70:P = 3
3105  FOR I = 1 TO 5: HTAB 8
3110  PRINT I;"- ";M$(I,3);: HTAB 16: PRINT "==>";:
       HTAB 20: PRINT M$(I,1)
3115  NEXT I
3120  VTAB 12: HTAB 8: PRINT "0- EXIT SECTION"
3125  VTAB 14: CALL  - 958: GOSUB 55
3130  IF N = 0 THEN  RETURN
3135  IF N > 0 AND N < 6 THEN 3145
3140  GOSUB 60: GOTO 3125
3145  GOSUB 3200
3150  GOSUB 100
3160  IF SW = 1 THEN 3125
3195  RETURN
3200  GOSUB 85: INPUT " = ";X
3210  GOSUB 360
3220  VTAB 16: PRINT X;" ";M$(N,3);"S = ";
       Y;" ";M$(N,1);"S"
3240  VTAB 18: PRINT X;" ";M$(N,3);"S = ";
       Z;" ";M$(N,2);"S"
3245  RETURN
3600  GOSUB 70:P = 1
3605  FOR I = 1 TO 5: HTAB 8
3610  PRINT I;"- ";M$(I,1);: HTAB 17: PRINT "==>";:
       HTAB 21: PRINT M$(I,3)
3615  NEXT I
3620  VTAB 12: HTAB 8: PRINT "0- EXIT SECTION"
3625  VTAB 14: CALL  - 958: GOSUB 55
3630  IF N = 0 THEN  RETURN
3635  IF N > 0 AND N < 6 THEN 3645
3640  GOSUB 60: GOTO 3625
3645  GOSUB 3700
3650  GOSUB 100
3660  IF SW = 1 THEN 3625
```

```
3695  RETURN
3700  GOSUB 85: INPUT " = ";X
3710  GOSUB 320
3720  VTAB 16: PRINT X;" ";M$(N,1);"S = ";
       Y;" ";M$(N,3);"S"
3740  VTAB 18: PRINT X;" ";M$(N,1);"S = ";
       Z;" ";M$(N,2);"S"
3745  RETURN
4000  HOME : PRINT : PRINT  TAB( 14)"*** AREA
       ***": PRINT
4010  GOSUB 200
4025  VTAB 12: CALL  - 958: GOSUB 55
4030  IF N = 0 THEN  RETURN
4035  IF N > 0 AND N < 3 THEN 4050
4040  GOSUB 60: GOTO 4025
4050  ON N GOSUB 4100,4600
4060  GOTO 4000
4100  GOSUB 70:P = 3
4105  FOR I = 1 TO 3: HTAB 8
4110  PRINT I;"- ";A$(I,3);: HTAB 16: PRINT "==>";:
       HTAB 20: PRINT A$(I,1)
4115  NEXT I
4120  VTAB 12: HTAB 8: PRINT "0- EXIT SECTION"
4125  VTAB 14: CALL  - 958: GOSUB 55
4130  IF N = 0 THEN  RETURN
4135  IF N > 0 AND N < 4 THEN 4145
4140  GOSUB 60: GOTO 4125
4145  GOSUB 4200
4150  GOSUB 100
4160  IF SW = 1 THEN 4100
4195  RETURN
4200  GOSUB 90: INPUT " = ";X
4210  GOSUB 370
4220  VTAB 16: PRINT X;" ";A$(N,3);" = ";Y;" ";A$(N,1)
4240  VTAB 18: PRINT X;" ";A$(N,3);" = ";Z;" ";A$(N,2)
4245  RETURN
4600  GOSUB 70:P = 1
4605  FOR I = 1 TO 3: HTAB 8
4610  PRINT I;"- ";A$(I,1);: HTAB 16: PRINT "==>";:
       HTAB 20: PRINT A$(I,3)
4615  NEXT I
4620  VTAB 12: HTAB 8: PRINT "0- EXIT SECTION"
4625  VTAB 14: CALL  - 958: GOSUB 55
4630  IF N = 0 THEN  RETURN
4635  IF N > 0 AND N < 4 THEN 4645
4640  GOSUB 60: GOTO 4625
4645  GOSUB 4700
4650  GOSUB 100
```

```
4660   IF SW = 1 THEN 4600
4695   RETURN
4700   GOSUB 90: INPUT " = ";X
4710   GOSUB 330
4720   VTAB 16: PRINT X;" ";A$(N,1);" = ";Y;" ";A$(N,3)
4740   VTAB 18: PRINT X;" ";A$(N,1);" = ";Z;" ";A$(N,2)
4745   RETURN
```

圖11-12　公制的轉換程式

11.3　遊戲程式

遊戲程式一直是電腦使用者之間引起爭論的主角，這種爭論的起因是因爲大多數人們認爲遊戲只是浪費時間而沒有任何實際的作用。

玩遊戲是一件浪費時間的事，但是我們總是需要休息的，是不！因此，何不找一些好的遊戲來試試看呢？

在電腦科學的領域內，我們也經常使用遊戲程式來做一些實驗，它們幫助了許多研究的工作；使用它們來做測試工作的原因是非常簡單的，遊戲似乎都具有下列的特性，這些特性都對某些實際的測試工作產生極大的功用。

* 一必須依照一定的規則去玩。
* 一某些限制的條件必須遵守。
* 一解決某些遊戲必須使用邏輯思考力。
* 一分數表示出了成功的次數。
* 一直接的聯繫。

以上的每一個項目都必須由程式設計師與操作員集中注意力去解決。

11.3.1 APPLE ALLEY遊戲

```
  10 REM **** APPLE ALLEY **************
  20 REM * A COMPUTER PROGRAM          *
  30 REM * ESPECIALLY FOR THE APPLE II *
  40 REM * WRITTEN BY: J.L. CAMPBELL   *
  50 REM * DATED: 1/5/78               *
  60 REM *                             *
  70 REM ******************************
  72 POKE 770,160:POKE 771,14:POKE 772,162:POKE
773,0:POKE 774,138
  74 POKE 775,24:POKE 776,233:POKE 777,1
  76 POKE 778,208:POKE 779,252
  80 POKE 780,141:POKE 781,48:POKE 782,192:POKE
783,232:POKE 784,224
  82 POKE 785,140:POKE 786,208:POKE 787,242:POKE
788,136:POKE 789,208
  85 POKE 790,237:POKE 791,96:POKE 792,0
  90 DIM A(110),C(10),ROLL$(4),ANS$(3),S(110),Z(10)
 100 L = 0:CALL -936:VTAB 5:TAB 4
 110 PRINT "*** WELCOME TO APPLE ALLEY ***"
 120 FOR I = 1 TO 2000:NEXT I
 140 CALL -936:VTAB 5:TAB 10
 150 PRINT "APPLE ALLEY BOWLING"
 155 PRINT:TAB 18:PRINT "BY"
 156 PRINT:TAB 13:PRINT "J.L CAMPBELL"
 160 VTAB 15:PRINT "WANT INSTRUCTIONS?(Y OR N)"
 170 INPUT ANS$
 180 IF ANS$="N" THEN 300
 200 CALL -936:VTAB 2:PRINT "****  APPLE ALLEY BOWLING
****"
 205 PRINT:PRINT "IN THIS BOWLING GAME THE APPLE II"
 210 PRINT "WILL KEEP SCORE FOR YOU AND UP TO"
 215 PRINT "A TOTAL OF TEN BOWLERS. YOU WILL BE"
 220 PRINT "PLAYING TEN FRAMES. ON THE PIN DIAGRAM,"
 225 PRINT "A '+' MEANS THE PIN IS STANDING, AND"
 230 PRINT "A '0' MEANS THE PIN IS DOWN. THREE "
 235 PRINT "PIN DIAGRAMS ARE SHOWN. THESE ARE:"
 240 PRINT "  1. STARTING SETUP"
 245 PRINT "  2. PINS AFTER FIRST BALL"
 250 PRINT "  3. PINS AFTER SECOND BALL"
 255 PRINT:PRINT "AFTER THE GAME, THE APPLE SCOREBOARD"
 260 PRINT "WILL SHOW THE SCORES OF ALL PLAYERS."
```

```
265 PRINT:PRINT "ENJOY THE BOWLING."
270 INPUT "HIT 'RETURN' TO CONTINUE.",ANS$
300 CALL -936:VTAB 10
310 INPUT "HOW MANY BOWLERS? ",R
330 PRINT:PRINT "THANK YOU...I'M SETTING PINS"
340 FOR I = 1 TO 110:A(I) = 0:S(I) = 0:NEXT I
350 FOR I = 1 TO 10:Z(I) = 0:NEXT I
370 F = 1
380 FOR P = 1 TO R
390 B = 1:T = 0
400 GOSUB 1000
405 CALL -936:VTAB 2:PRINT "***** APPLE ALLEY BOWLING
*****":PRINT
410 GOSUB 1150
412 POKE 771,10:POKE 785,100:CALL 770
415 VTAB 12
416 PRINT "READY!"
420 PRINT:PRINT "BOWLER ";P;" ROLL BALL ";B;" TYPE
ROLL"
430 INPUT ROLL$
450 GOSUB 1050
460 VTAB 4:T = T + 10
470 GOSUB 1150
490 REM ENTER SCORING ALGORITHM
500 GOSUB 1300
525 GOSUB 1700
540 IF UP > 0 AND UP < = 10 THEN GOSUB 1400
550 IF UP = 0 THEN GOSUB 1500
555 GOSUB 1600
556 IF UP = 0 THEN 682
560 B = 2:T = T + 10
570 VTAB 15:PRINT "ROLL SECOND BALL"
580 INPUT ROLL$
590 GOSUB 1050
600 VTAB 4:UP = 0
610 GOSUB 1150
620 REM ENTER FINAL SCORING
630 GOSUB 1300
650 GOSUB 1700
670 IF UP > 0 AND UP < = 10 THEN GOSUB 1400
680 IF UP = 0 THEN GOSUB 1800
681 GOSUB 1600
682 FOR J = 1 TO 1000:NEXT J
686 IF F = 10 THEN GOSUB 1900
690 NEXT P
700 F = F + 1
710 IF F < 11 THEN 380
720 REM SCORING RESULTS
```

```
 730 CALL -936:VTAB 2
 740 PRINT "*** APPLE ALLEY SCOREBOARD ANALYSIS ***"
 745 PRINT
 750 PRINT "FRAME 1  2  3  4  5  6  7  8 9 10"
 760 PRINT "- - - - - - - - - - - - - - - - - - - - -"
 770 FOR I = 1 TO R
 780 PRINT "  ";I;"  ";
 785 T = 4:TAB T
 790 FOR J = 1 TO 10
 800 TAB T + 3 * J
 810 PRINT A (I * 10 + J);
 820 NEXT J
 830 TAB 36:PRINT "=";Z (I)
 840 NEXT I
 850 PRINT:PRINT: INPUT "ANOTHER GAME? (Y OR N) ",ANS$
 870 L = L + 1
 880 IF ANS$ = "Y" THEN 330
 890 PRINT:PRINT "HOPE YOU ENJOYED THE BOWLING."
 900 PRINT:PRINT "YOU BOWLED ";L;" GAMES!"
 910 PRINT:PRINT "PLEASE PAY $";L * R;".00 TO THE ALLEY
OPERATOR!"
 920 END
 997 REM **
 998 REM ** INIT ARRAY C
 999 REM **
1000 FOR I = 1 TO 10
1010 C (I) = 1
1020 NEXT I
1030 RETURN
1047 REM **
1048 REM ** PIN FALL
1049 REM **
1050 FOR I = 1 TO 10
1060 IF (B = 1 OR B = 3) THEN X = RND (2) * RND (2)
1065 IF (B = 2 OR B = 4) THEN X1 = RND (2) * RND (2) *
RND (2)
1070 IF (B = 1 OR B = 3) THEN C (I) = X
1080 IF ((B = 2 OR B = 4) AND C (I) = 1) THEN C (I) =
X1
1100 NEXT I
1110 RETURN
1147 REM **
1148 REM ** PIN DIAGRAM
1149 REM **
1150 VTAB 4:TAB 1
1151 PRINT "BOWLER ";P;" ..FRAME ";F;" ..BALL ";B;"
TOTALS"
```

```
1155 PRINT
1160 K = 1:VTAB 6
1170 FOR I = 1 TO 4
1180 TAB I + T
1190 FOR J = 1 TO 5-I
1200 IF C(K) = 0 THEN 1230
1210 PRINT "+ ";
1220 GOTO 1240
1230 PRINT "0 ";
1240 K = K + 1
1250 NEXT J:PRINT
1270 NEXT I:PRINT
1290 RETURN
1297 REM **
1298 REM ** COUNT PINS STANDING
1299 REM **
1300 UP = 0
1310 FOR I = 1 TO 10
1320 UP = UP + C (I)
1330 NEXT I
1340 RETURN
1397 REM **
1398 REM ** 1 TO 10 PINS STANDING
1399 REM **
1400 IF B = 2 THEN 1440
1405 IF (B = 3 OR B = 4) THEN 1491
1410 IF S(P * 10 + F -1) = 2 THEN A (P * 10 + F -1) = A
(P * 10 + F -1) + 10 - UP
1420 IF S(P * 10 + F -1) = 2 THEN Z (P) = Z (P) + 10 -
UP
1425 IF (S(P * 10 + F -2) = 3 AND S(P * 10 + F -1) = 3)
THEN A(P * 10 + F -2) = A(P * 10 + F -2) + 10 - UP
1430 IF (S(P * 10 + F -2) = 3 AND S(P * 10 + F -1) = 3)
THEN Z(P) = Z(P) + 10 - UP
1431 RETURN
1440 A(P * 10 + F) = A(P * 10 + F) + 10 - UP
1450 Z(P) = Z (P) + 10 - UP
1460 S(P * 10 + F) = 0
1470 IF S(P * 10 + F - 1) = 3 THEN A(P * 10 + F - 1) =
A(P * 10 + F - 1) + 10 - UP
1480 IF S(P * 10 + F - 1) = 3 THEN Z(P) = Z(P) + 10 -
UP
1485 IF F = 11 THEN A(P * 10 + F - 1) = A(P * 10 + F -
1) + 10 - UP
1490 RETURN
1491 Z(P) = Z(P) + 10 - UP
```

```
1492 A(P * 10 + F) = A(P * 10 + F) + 10 - UP
1493 RETURN
1497 REM *
1498 REM * STRIKE BALL
1499 REM *
1500 IF (B = 3 OR B = 4) THEN 1565
1510 A(P * 10 + F) = A(P * 10 + F) + 10
1515 Z(P) = Z(P) + 10
1520 S(P * 10 + F) = 3
1530 IF (S(P * 10 + F -1) = 3 AND S (P * 10 + F -2) =
3) THEN A(P * 10 + F - 2) = A(P * 10 + F - 2) + 10
1535 IF (S(P * 10 + F -1) = 3 AND S(P * 10 + F -2) = 3)
THEN Z(P) = Z(P) + 10
1540 IF S(P * 10 + F -1) = 3 THEN A(P * 10 + F -1) =
A(P * 10 + F -1) + 10
1545 IF S(P * 10 + F -1) = 3 THEN Z(P) = Z(P) + 10
1550 IF S(P * 10 + F -1) = 2 THEN A(P * 10 + F -1) = A
(P * 10 + F -1) + 10
1555 IF S(P * 10 + F -1) = 2 THEN Z(P) = Z(P) + 10
1560 RETURN
1565 A(P * 10 + F) = A(P * 10 + F) + 10
1570 Z(P) = Z(P) + 10
1575 IF S(P * 10 + F -1) = 3 THEN A(P * 10 + F -1) =
A(P * 10 + F -1) + 10
1580 IF S(P * 10 + F -1) = 3 THEN Z(P) = Z(P) + 10
1585 RETURN
1597 REM *
1598 REM * TOTAL SCORES
1599 REM *
1600 FOR I = 1 TO R
1610 VTAB 5 + I: TAB 33
1620 PRINT I;" ";Z(I)
1630 NEXT I
1640 RETURN
1697 REM *
1698 REM * ANNOTATE EACH ROLL
1699 REM *
1700 VTAB 12: TAB T
1701 IF B = 1 THEN 1704
1702 IF B = 2 THEN 1740
1704 IF UP = 0 THEN 1710
1706 IF UP > 0 AND UP < 10 THEN 1720
1708 IF UP = 10 THEN 1730
1710 PRINT "STRIKE!!"
1712 POKE 771,30:POKE 785,225:CALL 770
1714 FOR J = 1 TO 1000
1716 NEXT J
1718 RETURN
```

```
1720 PRINT "OH WELL!"
1722 POKE 771,UP:POKE 785,200:CALL 770
1724 RETURN
1730 PRINT "GUTTER!!"
1732 POKE 771,UP:POKE 785,200:CALL 770
1734 RETURN
1740 IF UP = 0 THEN 1750
1742 IF UP > 0 AND UP < 10 THEN 1746
1744 IF UP = 10 THEN 1730
1746 PRINT "* OPEN *"
1747 POKE 771,UP:POKE 785,235:CALL 770
1748 RETURN
1750 PRINT "SPARE!!!"
1752 POKE 771,10:POKE 785, RND (11) * 10 +120:CALL 770
1754 RETURN
1797 REM *
1798 REM * SPARE BALL
1799 REM *
1800 A(P * 10 + F) = A(P * 10 + F) + 10
1810 S(P * 10 + F) = 2
1820 IF (S(P * 10 + F -1) = 3 AND B <> 4) THEN A(P * 10
+ F -1) = A(P * 10 + F -1) + 10
1830 IF (S(P * 10 + F -1) = 3 AND B <> 4) THEN Z(P) =
Z(P) + 10
1840 Z(P) = Z(P) + 10
1890 RETURN
1897 REM *
1898 REM * EXTRA BALLS
1899 REM *
1900 IF S(P * 10 + F) = 0 THEN RETURN
1902 FOR I = 0 TO 6: VTAB 6 + I: TAB 10:PRINT "
":NEXT I
1920 B = 3:T = 0:V = 17
1930 GOSUB 1000
1940 VTAB 4
1950 GOSUB 1150
1960 VTAB V:PRINT "ROLL EXTRA BALL"
1970 INPUT ROLL$
1980 GOSUB 1050
1990 T = T + 10:VTAB 4:TAB T
2000 GOSUB 1150
2002 FOR I = 1 TO 1000
2004 NEXT I
2010 GOSUB 1300
2020 GOSUB 1700
2050 IF UP > 0 AND UP <= 10 THEN GOSUB 1400
```

```
2060 IF B = 3 AND UP = 0 THEN GOSUB 1500
2065 IF B = 4 AND UP = 0 THEN GOSUB 1800
2070 S(P * 10) = 2 THEN RETURN
2080 B = 4
2090 V = V + 1
2100 VTAB V:PRINT "ROLL SECOND BALL"
2120 INPUT ROLL$
2130 S(P * 10 + F) = 2
2140 GOTO 1980
```

圖11-13 APPLE ALLEY程式

11.3.2 NIM GAME遊戲

```
1   REM *** NIM GAME *******
2   REM *
3   REM * WRITTEN BY:JL CAMPBELL
4   REM *        DATED:07/27/80
5   REM *
6   REM * COMPUTER:APPLE II
7   REM * LANGUAGE:APPLESOFT II
8   REM *
9   REM ******************
10   TEXT : CLEAR : POKE 216,0:A =  FRE (0)
20   DIM X(50)
22   FOR I = 1 TO 50:X(I) = 0: NEXT I
40   HOME : VTAB (10): INPUT "WANT INSTRUCTIONS?(Y/N)
";A$
45   IF  LEFT$ (A$,1) = "Y" THEN  GOSUB 5000
50   GOTO 900
51   REM *************
52   REM * GEN'L SUBR *
53   REM *************
60   VTAB (23): HTAB 10: INVERSE : PRINT "ILLEGAL
ENTRY": NORMAL : PRINT  CHR$ (7)
65   FOR L = 1 TO 1000: NEXT L: RETURN
70   HOME : PRINT : PRINT  TAB( 12)"*** NIM GAME ***":
PRINT : RETURN
99   REM * PLOT BOARD
100   GR
105   FOR I = 1 TO 5
110   FOR J = 1 TO 10
115   COLOR=  RND (1) * 14 + 1
120   IF X(10 * (I - 1) + J) = 0 THEN 140
```

```
125  FOR K = 0 TO 2
130  HLIN 4 * (J - 1),4 * (J - 1) + 2 AT 4 * (I - 1) +
K + 12
135  NEXT K
140  NEXT J,I
145  RETURN
699  REM * HUMAN MOVES
700  GOSUB 70
705  PRINT "PLAYER #";P;: INPUT " REMOVES HOW MANY? =
";N
710  IF N > 0 AND N < 4 THEN 720
715 : GOSUB 60: GOTO 700
720 Z = NR - N
725  IF Z <  = 0 THEN SW = 1: RETURN
730  IF Z > 0 THEN  RETURN
735  GOSUB 60: GOTO 700
799  REM * COMPUTER MOVES
800 Q = NR / 4
805 R = Q -  INT (NR / 4)
815 R1 = R + 1
820 N = 4 - R1
825  IF N = 0 THEN N = 3
830 Z = NR - N
835  IF Z <  = 0 THEN SW = 1: RETURN
840  IF Z > 0 THEN  RETURN
845  RETURN
899  REM * MAIN DRIVER
900  GOSUB 70
905  INPUT "HOW MANY TILES?(MAX=50) ";NR
910  FOR I = 1 TO NR:X(I) = 1: NEXT I
915  FOR P = 1 TO 2
920  GOSUB 100
925 SW = 0
930  ON P GOSUB 700,800
935  IF SW = 1 THEN 1000
975 NR = NR - N
980  FOR I = NR + 1 TO 50:X(I) = 0: NEXT I
990  NEXT P
995  GOTO 915
1000  REM * END PLAY
1010  HOME : VTAB (21): PRINT "PLAYER ";3 - P;"
WINS!!!"
1020  INPUT "ANOTHER GAME?(Y/N) ";A$
1030  IF  LEFT$ (A$,1) = "Y" THEN  TEXT : HOME : GOTO
900
1040  TEXT : HOME : VTAB (10)
```

```
1050  PRINT "GOODBYE FOR NOW...": END
4999  REM * INTSRUCTIONS
5000  GOSUB 70
5010  PRINT "THIS LOGIC GAME, KNOWN BY MANY NAMES,"
5020  PRINT "SUCH AS 23 MATCHES, 23 BRICKS, NIM,"
5030  PRINT "ETC, IS A CLASSIC TWO PERSON EXERCISE"
5040  PRINT "IN LOGIC. THERE EXISTS MANY VARIATIONS"
5050  PRINT "OF THIS GAME.": PRINT
5060  PRINT "THE OBJECT IS TO REMOVE TILES FROM THE"
5070  PRINT "PILE, 1,2, OR 3, IN SUCH A WAY, THAT"
5080  PRINT "YOUR OPPONENT MUST TAKE THE LAST TILE."
5090  PRINT "EACH PLAYER MUST REMOVE AT LEAST 1 TILE"
5100  PRINT "AND NOT MORE THAN 3 TILES.": PRINT
5110  PRINT "THE COMPUTER IS YOUR OPPONENT!": PRINT
5120  PRINT "PLEASE DON'T CHEAT.": PRINT
5130  PRINT "GOOD LUCK!"
5140  VTAB (22): INPUT "TYPE RETUN KEY TO CONTINUE...
";A$
5150  RETURN
```

圖11-14 NIM Game程式

11.3.3 QUBE FOUR 遊戲

```
1  REM  **** QUBE FOUR ****
2  REM  *
3  REM  * WRITTEN BY:JL CAMPBELL
4  REM  *            MA LANGSTON
5  REM  *
6  REM  *   COMPUTER:APPLE II
7  REM  *   LANGUAGE:APPLESOFT
8  REM  *      DATED:11/28/78
9  REM  *
10  REM  *******************
20  GOTO 2500
30  REM  * PLOT BEADS
32  FOR I = 1 TO 4: FOR J = 1 TO 4: FOR K = 1 TO 4
34  IF A(I,J,K) =  - 1 THEN 48
36  IF A(I,J,K) = 1 THEN 42
38  COLOR= 2: GOTO 44
42  COLOR= 4
44  HLIN 10 * (J - 1) + 2,10 * (J - 1) + 7 AT 10 * (I -
1) + 9 - (2 * K - 1)
48  NEXT K,J,I
49  RETURN
50  REM  * PLOT THE BOARD
52  GR : COLOR= 13
54  FOR I = 0 TO 30 STEP 10
56  HLIN 0,39 AT I: VLIN 0,39 AT I
58  HLIN 0,39 AT I + 9: VLIN 0,39 AT I + 9
60  NEXT I
62  COLOR= 9
64  FOR I = 4 TO 34 STEP 10: FOR J = 1 TO 31 STEP 10
66  VLIN J + 1,J + 7 AT I: VLIN J + 1,J + 7 AT I + 1
68  NEXT J,I: RETURN
200  REM  * TWO-PLY SEARCH SUBR
202 C = 0
204 X = 1
206 Y = 1
208 Z = 1
210  IF A(X,Y,Z) =  - 1 THEN 225
215 Z = Z + 1: IF Z <  = 4 THEN 210
225  GOSUB 500
230  IF C = 1 THEN 240
235 V0(X,Y) = 2 * D: GOTO 245
240 V1(X,Y) = 10 + 2 * D
245 Y = Y + 1
250  IF Y <  = 4 THEN 208
255 X = X + 1
```

```
260  IF X <  = 4 THEN 206
265 C = C + 1
270  IF C <  = 1 THEN 204
275  REM  * SELECT BEST PLAY
280 C = 0:E = 0
285 XX = 1
290 YY = 1
295  IF C = 1 THEN 320
300  IF V0(XX,YY) < E THEN 335
302  IF V0(XX,YY) = E AND  RND (1) <  = 0.6 THEN 335
305 X = XX:Y = YY
310 E = V0(XX,YY)
315  GOTO 335
320  IF V1(XX,YY) < E THEN 335
322  IF V1(XX,YY) = E AND  RND (1) <  = 0.6 THEN 335
325 X = XX:Y = YY
330 E = V1(XX,YY)
335 YY = YY + 1
340  IF YY <  = 4 THEN 295
345 XX = XX + 1
350  IF XX <  = 4 THEN 290
355 C = C + 1
360  IF C <  = 1 THEN 285
365 Z = 1
370  IF A(X,Y,Z) =  - 1 THEN 385
375 Z = Z + 1
380  GOTO 370
385  REM. * NO WIN-CHECK OPPONENTS NEW OPENING
390  IF E >  = 70 THEN  RETURN
395 C = 0:Z = Z + 1
400  GOSUB 500
405 Z = Z - 1
410  IF D < 30 THEN  RETURN
415 V0(X,Y) = 0:V1(X,Y) = 0
420  GOTO 275
425  RETURN
500  REM  * EVAL POTENTIAL PLAY
505 D = 0:V = 0:W = 0
510  IF Z > 4 THEN  RETURN
515  REM  * EXAMINE ORTHOGONALS
520  FOR I = 1 TO 4: IF A(I,Y,Z) =  - 1 THEN 530
522  IF A(I,Y,Z) <  > C THEN 534
525 V = V + 1
530  NEXT I
```

```
532 D = V
534 V = 0
540  FOR I = 1 TO 4: IF A(X,I,Z) =  - 1 THEN 550
542  IF A(X,I,Z) <  > C THEN 565
545 V = V + 1
550  NEXT I
555  IF D >  = V THEN 565
560 D = V
565 V = 0
570  FOR I = 1 TO 4: IF A(X,Y,I) =  - 1 THEN 580
572  IF A(X,Y,I) <  > C THEN 595
575 V = V + 1
580  NEXT I
585  IF D >  = V THEN 595
590 D = V
595 V = 0
600  REM  * EXAMINE Z-PLANE DIAG
605  IF X <  > Y THEN 645
610 W = W + 1
615  FOR I = 1 TO 4: IF A(I,I,Z) =  - 1 THEN 625
617  IF A(I,I,Z) <  > C THEN 640
620 V = V + 1
625  NEXT I
630  IF D >  = V THEN 640
635 D = V
640 V = 0
645  IF (X + Y) <  > 5 THEN 685
650 W = W + 1
655  FOR I = 1 TO 4: IF A(I,5 - I,Z) =  - 1 THEN 665
657  IF A(I,5 - I,Z) <  > C THEN 680
660 V = V + 1
665  NEXT I
670  IF D >  = V THEN 680
675 D = V
680 V = 0
685  REM  * EXAMINE Y-PLANE DIAG
690  IF X <  > Z THEN 730
695 W = W + 1
700  FOR I = 1 TO 4: IF A(I,Y,I) =  - 1 THEN 710
702  IF A(I,Y,I) <  > C THEN 725
705 V = V + 1
710  NEXT I
715  IF D >  = V THEN 725
720 D = V
725 V = 0
730  IF (X + Z) <  > 5 THEN 770
735 W = W + 1
```

```
740  FOR I = 1 TO 4: IF A(I,Y,5 - I) =  - 1 THEN 750
742  IF A(I,Y,5 - I) <  > C THEN 765
745 V = V + 1
750  NEXT I
755  IF D >  = V THEN 765
760 D = V
765 V = 0
770  REM  * EXAMINE X-PLANE DIAG
775  IF Y <  > Z THEN 815
780 W = W + 1
785  FOR I = 1 TO 4: IF A(X,I,I) =  - 1 THEN 795
787  IF A(X,I,I) <  > C THEN 810
790 V = V + 1
795  NEXT I
800  IF D >  = V THEN 810
805 D = V
810 V = 0
815  IF (Y + Z) <  > 5 THEN 855
820 W = W + 1
825  FOR I = 1 TO 4: IF A(X,I,5 - I) =  - 1 THEN 835
827  IF A(X,I,5 - I) <  > C THEN 850
830 V = V + 1
835  NEXT I
840  IF D >  = V THEN 850
845 D = V
850 V = 0
855  REM  * EXAMINE SPACE DIAG
860  IF X <  > Y THEN 905
865  IF (Y + Z) <  > 5 THEN 905
870 W = W + 1
875  FOR I = 1 TO 4: IF A(I,I,5 - I) =  - 1 THEN 885
877  IF A(I,I,5 - I) <  > C THEN 900
880 V = V + 1
885  NEXT I
890  IF D >  = V THEN 900
895 D = V
900 V = 0
905  IF X <  > Z THEN 950
910  IF (Y + Z) <  > 5 THEN 950
915 W = W + 1
920  FOR I = 1 TO 4: IF A(I,5 - I,I) =  - 1 THEN 930
922  IF A(I,5 - I,I) <  > C THEN 945
925 V = V + 1
930  NEXT I
935  IF D >  = V THEN 945
940 D = V
```

```
945 V = 0
950  IF Y <  > Z THEN 995
955  IF (X + Y) <  > 5 THEN 995
960 W = W + 1
965  FOR I = 1 TO 4: IF A(5 - I,I,I) =  - 1 THEN 975
967  IF A(5 - I,I,I) <  > C THEN 990
970 V = V + 1
975  NEXT I
980  IF D >  = V THEN 990
985 D = V
990 V = 0
995  IF X <  > Y THEN 1035
1000  IF X <  > Z THEN 1035
1005 W = W + 1
1010  FOR I = 1 TO 4: IF A(I,I,I) =  - 1 THEN 1020
1012  IF A(I,I,I) <  > C THEN 1035
1015 V = V + 1
1020  NEXT I
1025  IF D >  = V THEN 1035
1030 D = V
1035 D = W + (10 * D)
1040  RETURN
1100  PRINT "I'M THINKING - QUIET PLEASE . . .": RETURN

1110  PRINT "JUST A MINUTE, LET ME PICK MY MOVE":
RETURN
1120  PRINT "MY TURN %&*@ $# - KEEP AWAY FROM KEYS!":
RETURN
1130  PRINT "BOXCARS COMING UP--OOPS! WRONG GAME!!!":
RETURN
1140  PRINT "I'VE GOT YOU NOW - WANT TO GIVE UP?":
RETURN
1150  PRINT "WATCH OUT FOR DIAGONALS ON THIS MOVE.":
RETURN
1160  PRINT "HURRAH!! IT'S FINALLY MY TURN!!": RETURN
1170  PRINT "DON'T CATCH ME CHEATING ON THIS PLAY.":
RETURN
1180  PRINT "I'LL BET YOU CAN'T GUESS MY NEXT PLAY.":
RETURN
1190  PRINT "ARE YOU SURE YOU KNOW HOW TO PLAY?":
RETURN
1200  PRINT  TAB( 10)"**** QUBE-FOUR ****": PRINT :
RETURN
1210  VTAB 22: INPUT "HIT RETURN TO CONTINUE.";ANS$:
RETURN
1220  VTAB 22: PRINT "SELECT YOUR ROW,COLUMN MOVE ";:
INPUT X,Y: RETURN
1230  HOME : VTAB 21: PRINT "NOW YOU KNOW THAT MOVE IS
ILLEGAL!!"
1240  PRINT "PLAY HONESTLY!!"
```

```
1250  FOR L = 1 TO 2000: NEXT L: RETURN
1260  HOME : VTAB 21: PRINT "THERE ARE ALREADY FOUR
BEADS ON PEG ";X;",";Y: GOTO 1250
2500  TEXT : HOME : GOSUB 1200
2502  DIM ANS$(6),A(4,4,4)
2504  DIM V0(4,4),V1(4,4)
2510  FOR I = 1 TO 4: FOR J = 1 TO 4: FOR K = 1 TO
4:A(I,J,K) =  - 1: NEXT K,J,I
2512  PRINT "THE OBJECT IS TO BE THE FIRST PLAYER"
2514  PRINT "TO PLACE FOUR BEADS IN A ROW EITHER"
2516  PRINT "HORIZONTALLY, VERTICALLY, OR ON ANY"
2518  PRINT "OF THE DIAGONALS."
2520  PRINT : PRINT "THE PLAY COMMENCES BY RESPONDING
TO THE"
2522  PRINT "PROMPTING MESSAGE AT THE BOTTOM OF THE"
2524  PRINT "SCREEN. YOU ENTER YOUR CHOICE WHEN"
2526  PRINT "PROMPTED BY ENTERING FIRST THE ROW"
2528  PRINT "AND THEN THE COLUMN SEPARATED BY A"
2530  PRINT "COMMA. THE ROWS ARE NUMBERED 1-4 FROM"
2532  PRINT "TOP TO BOTTOM; THE COLUMNS ARE NUMBERED"
2534  PRINT "1-4 FROM LEFT TO RIGHT."
2536  PRINT : PRINT "FOR EXAMPLE, THE PLAY 1,4 WILL
PLACE A"
2538  PRINT "BEAD OF YOUR COLOR IN THE UPPER RIGHT"
2540  PRINT "CORNER OF THE BOARD.  THE COMPUTER WILL":
PRINT "COMPUTE THE CORRECT BEAD HEIGHT."
2542  GOSUB 1210: HOME : GOSUB 1200
2548  PRINT "PLEASE WAIT FOR THE PROMPTING MESSAGES!"
2550  PRINT "IF YOU DON'T, I MAY GO BANANAS!!!!"
2552  PRINT : PRINT "ONE,TWO OR FOUR PLAYERS MAY PLAY.
"
2554  PRINT "WITH ONE PLAYER, THE COMPUTER IS YOUR"
2556  PRINT "OPPONENT. TWO PLAYERS MAY PLAY REGULAR"
2558  PRINT "QUBE-FOUR. FOUR PLAYERS MAY PLAY"
2560  PRINT "PARTNERS IN QUBE-FOUR."
2562  VTAB 13: PRINT "HOW MANY PLAYERS?"
2564  INPUT "ANSWER WITH 1,2, OR 4.  ";NP
2566  IF NP = 1 THEN 2980
2568  IF NP = 2 OR NP = 4 THEN 2600
2570  HOME : GOSUB 1200: PRINT "I SAID 1,2, OR 4.
ANSWER CORRECTLY": PRINT "OR I MAY QUIT.": GOTO 2562
2600  GOSUB 50:T = 2:P = 2
2610  IF T = 1 THEN 2630
2620 P = 3 - P
2630 T = 3 - T
2640 C = T - 1
```

```
2650  FOR L = 1 TO 3: CALL  - 1052: NEXT L
2660  IF NP = 2 THEN 2680
2670  HOME : VTAB 21: PRINT "TEAM ";T;" PLAYER ";P:
GOTO 2690
2680  HOME : VTAB 21: PRINT "PLAYER ";T
2690  GOSUB 1220
2700  IF X < 1 OR X > 4 THEN 2730
2710  IF Y < 1 OR Y > 4 THEN 2730
2720  GOTO 2750
2730  GOSUB 1230: GOTO 2660
2750  FOR I = 1 TO 4
2752 Z = I: IF A(X,Y,Z) =  - 1 THEN 2760
2754  NEXT I
2756  GOSUB 1260
2758  GOSUB 1250
2760 A(X,Y,Z) = C: GOSUB 30
2770  GOSUB 500
2780  IF D < 40 THEN 2610
2785  HOME
2790  IF NP = 2 THEN 2820
2800  PRINT "CONGRATULATIONS - TEAM ";T;" WINS!!"
2810  END
2820  PRINT "CONGRATULATIONS - PLAYER ";T;" WINS!!!"
2830  END
2980  HOME : VTAB 10: PRINT "DO YOU WISH TO PLAY
FIRST?": INPUT "ANSWER EITHER YES OR NO. ";ANS$
2990  GOSUB 50: IF ANS$ = "NO" THEN 3190
3000  REM  * MAIN DRIVER PROGRAM
3005  FOR L = 1 TO 3: CALL  - 1052: NEXT L
3010  HOME : GOSUB 1220
3020  IF X < 1 OR X > 4 THEN 3070
3030  IF Y < 1 OR Y > 4 THEN 3070
3040  GOTO 3080
3070  GOSUB 1230: GOTO 3010
3080  FOR I = 1 TO 4
3090 Z = I: IF A(X,Y,Z) =  - 1 THEN 3110
3100  NEXT I
3105  GOSUB 1260: GOTO 3070
3110 A(X,Y,Z) = 0
3120  GOSUB 30: REM  * PLOT BOARD
3130 C = 0
3140  GOSUB 500
3150  IF D < 40 THEN 3190
3160  HOME : VTAB 21: PRINT "CONGRATULATIONS!!! YOU
HAVE WON."
3170  PRINT "I'M NO MATCH FOR YOUR SKILL!"
3180  END
```

```
3190  REM  * APPLE II PLAYS BOARD
3195  HOME : VTAB 22: ON 1 +  INT (10 *  RND (1)) GOSUB
1100,1110,1120,1130,1140,1150,1160,1170,1180,1190
3200  GOSUB 200
3210 A(X,Y,Z) = 1
3215  FOR L = 1 TO 5: CALL  - 1052: NEXT L
3220  HOME : VTAB 22: PRINT "I WILL PLAY ROW= ";X;"
COL= ";Y
3230  GOSUB 30:C = 1
3235  FOR L = 1 TO 1500: NEXT L
3240  GOSUB 500
3250  IF D < 40 THEN 3010
3260  HOME : VTAB 21: PRINT "I'M SORRY, LOOKS LIKE YOU
LOSE."
3270  PRINT "BETTER LUCK NEXT TIME!!"
3280  END
```

圖11-15 Qubu four程式

11.3.4 REVERSE程式

```
1  REM *** REVERSE ***
2  REM *
3  REM * WRITTEN BY:JL CAMPBELL
4  REM *       DATED:07/19/80
5  REM *
6  REM * COMPUTER:APPLE II
7  REM * LANGUAGE:APPLESOFT II
8  REM *
9  REM ********************
10  TEXT : CLEAR : POKE 216,0:A =  FRE (0)
20  DIM X(9),Y(9)
30  HOME : PRINT : PRINT  TAB( 12)"*** REVERSE ***":
PRINT
35  VTAB (4): CALL  - 958: INPUT "WANT
INSTRUCTIONS?(Y/N) ";A$
40  IF  LEFT$ (A$,1) = "Y" THEN  GOSUB 2000
50  GOTO 900
51  REM **************
52  REM * GEN'L SUBR *
53  REM **************
60  VTAB (20): HTAB 10: INVERSE : PRINT "ILLEGAL
     ENTRY": NORMAL : PRINT  CHR$ (7)
65  FOR L = 1 TO 1000: NEXT L: RETURN
100  VTAB (6): CALL  - 958: HTAB 5
105  FOR I = 1 TO 9: PRINT X(I);" ";: NEXT I:
      PRINT : RETURN
200  REM * RANDOM LIST
205  FOR I = 1 TO 9:Y(I) = I: NEXT I
210  FOR I = 1 TO 9
215 N =  INT ( RND (1) * 10)
220  IF Y(N) = 0 THEN 215
225 X(I) = Y(N):Y(N) = 0
230  NEXT I
295  RETURN
900  REM * MAIN DRIVER
925  GOSUB 200
930  VTAB (4): CALL  - 958:T = 0
935  PRINT "HERE IS THE LIST......
940  GOSUB 100
950  VTAB (8): CALL  - 958: INPUT "HOW MANY SHALL I
      REVERSE? ";N
955  IF N > 0 AND N < 10 THEN 970
960  GOSUB 60: GOTO 950
970 T = T + 1
```

```
975  REM * REVERSE #
980  FOR L = 1 TO  INT (N / 2)
985 Z = X(L)
990 X(L) = X(N - L + 1)
995 X(N - L + 1) = Z
1000  NEXT L
1005  REM * CHECK FOR WIN
1010  FOR L = 1 TO 9
1015  IF X(L) <  > L THEN 940
1020  NEXT L
1025  GOSUB 100
1030  PRINT : PRINT "YOU WON IN ";T;" MOVES!"
1040  PRINT : INPUT "TRY AGAIN?(Y/N) ";A$
1045  IF  LEFT$ (A$,1) = "Y" THEN 925
1050  PRINT : PRINT "GOODBYE FOR NOW...": END
2000  REM * INSTRUCTIONS
2010  VTAB (4): CALL  - 958
2020  PRINT "THE OBJECT OF REVERSE IS TO ARRANGE THE"
2030  PRINT "NUMBERS IN ASCENDING ORDER. YOU START"
2040  PRINT "WITH THEM IN RANDOM ORDER. YOU MUST"
2050  PRINT "TELL ME HOW MANY TO REVERSE. I WILL"
2060  PRINT "THEN REVERSE THAT MANY STARTING FROM"
2070  PRINT "LEFT. YOU KEEP REVERSING NUMBERS UNTIL"
2080  PRINT "THEY ARE IN ORDER. I WILL TELL YOU WHEN"
2090  PRINT "YOU HAVE WON.
2100  PRINT : PRINT "GOOD LUCK..."
2110  PRINT : INPUT "TYPE RETURN KEY TO CONTINUE
       ... ";A$
2120  RETURN
```

圖11-16 Reverse程式

11.3.5 生命週期 (BIORHYTHM)

```
1  REM  **** BIO CYCLE PROG ****
2  REM  *
3  REM  * COMPUTER:APPLE II
4  REM  * LANGUAGE:APPLESOFT II
5  REM  *
6  REM  *      DATED:10/25/79
7  REM  * WRITTEN BY:JL CAMPBELL
8  REM  *
9  REM  **************************
20  DIM M$(12),D$(7)
30  FOR I = 1 TO 12: READ M$(I): NEXT I
32  DATA  "JAN.","FEB.","MAR.","APR."
34  DATA  "MAY ","JUNE","JULY","AUG."
36  DATA  "SEP.","OCT.","NOV.","DEC."
40  FOR I = 1 TO 7: READ D$(I): NEXT I
42  DATA
"SUN.","MON.","TUE.","WED.","THUR","FRI.","SAT."
50  HOME : PRINT : PRINT  TAB( 10)"*** BIO-RHYTHM ***":
PRINT
100  REM  * BIO-CYCLE DRIVER
110  VTAB (4): CALL  - 958
120  INPUT "NAME          = ";A0$
130  INPUT "ADDRESS       = ";A1$
140  INPUT "CITY          = ";A2$
150  INPUT "STATE         = ";A3$
160  INPUT "ZIP CODE      = ";A4$
180 PG = 0:LN = 0
190  GOTO 1000
300  REM  PRINTER SUBR $ HEADINGS
310  HOME : VTAB (6): PRINT "DO THE FOLLOWING;"
320  PRINT : PRINT  TAB( 5)"1- LOAD PRINTER PAPER"
322  PRINT  TAB( 5)"2- TURN ON THE PRINTER"
324  PRINT  TAB( 5)"3- SET TOP OF FORM."
326  PRINT : INPUT "TYPE RETURN TO CONTINUE ";ANS$
```

```
328  PR# 1
330  PRINT  CHR$ (9);"132N"
340 PG = PG + 1
342  PRINT  TAB( 60)"PAGE ";PG: PRINT "PREPARED FOR:"
344  PRINT  TAB( 14)A0$
346  PRINT  TAB( 14)A1$
348  PRINT  TAB( 14)A2$;", ";A3$;" ";A4$
350  PRINT " "
355 LN = 8
360  RETURN
1000  VTAB (4): CALL  - 958: INPUT "ENTER BIRTH DATE AS
M,D,Y.";M,D,Y1
1020  GOSUB 1840
1040 J4 = J2:W1 = W
1050  VTAB (6): CALL  - 958
1060  PRINT "YOU WERE BORN ON A ";D$(W1);" -WHOPPEE!"
1080  VTAB (7): CALL  - 958
1100  VTAB (8): PRINT "ENTER MONTH,DAY,YEAR FOR THE
START OF": INPUT "YOUR BIO-RHYTHM. ";M,D,Y1
1130  GOSUB 1840
1150 J5 = J2:W1 = W
1170  VTAB (10): PRINT "HOW MANY DAYS DO YOU WANT YOUR
       CHART": INPUT "TO COVER? ";J7
1190 J6 = J7 + J5
1200  PRINT
1210  PRINT "WHICH WOULD YOU LIKE TO DO?"
1220  PRINT  TAB( 5)"1- PLOT YOUR BIORHYTHM."
1230  PRINT  TAB( 5)"2- LIST YOUR CRITICAL DAYS."
1240  PRINT : INPUT "WHICH ONE? ";Z
1250  IF Z < 3 AND Z > 0 THEN 1260
1252  PRINT : PRINT "INCORRECT ENTRY...TRY AGAIN!"
1254  FOR I = 1 TO 1500: NEXT I
1256  VTAB (13): CALL  - 958: GOTO 1210
1260  ON Z GOTO 2200,1310
1310  GOSUB 310
1355  PRINT " ","  THIS IS A LIST OF YOUR CRITICAL
       DAYS FOR THE PERIOD"
1360  PRINT " ","  BEGINNING ";M$(M);" ";D;", ";Y1
1370  PRINT " "
1380  PRINT " DAY","        DATE","  CYCLE   (0=CRITICAL
       DAY)"
1390  FOR I = 1 TO 72: PRINT "=";: NEXT I: PRINT
1400  FOR X = J5 TO (J5 + J7)
```

```
1410 J2 = X
1420  GOSUB 1960
1430 D9 = X - J4
1440 P = (D9 / 23) -  INT ((D9 / 23) + C9)
1450 P =  INT (23 * P + C9)
1460  IF P < 19 THEN 1480
1470 P = 23 - P
1480 E = (D9 / 28) -  INT ((D9 / 28) + C9)
1490 E =  INT (28 * E + C9)
1500  IF E < 19 THEN 1520
1510 E = 28 - E
1520 C = (D9 / 33) -  INT ((D9 / 33) + C9)
1530 C =  INT (33 * C + C9)
1540  IF C < 19 THEN 1560
1550 C = 33 - C
1560  REM  *
1570  IF P = 0 THEN 1660
1580  IF P = 11 THEN 1660
1590  REM  *
1600  IF E = 0 THEN 1690
1610  IF E = 14 THEN 1690
1620  REM  *
1630  IF C = 0 THEN 1720
1640  IF C = 16 THEN 1720
1650  GOTO 1740
1660  PRINT D$(W),M$(M1);" ";D1;", ";Y2;" ","
       PHYSICAL","0"
1670 F6 = F6 + 1
1680  GOTO 1590
1690  PRINT D$(W),M$(M1);" ";D1;", ";Y2;" ","
EMOTIONAL","0"
1700 F6 = F6 + 1
1710  GOTO 1620
1720  PRINT D$(W),M$(M1);" ";D1;", ";Y2;" ","
       MENTAL","0"
1730 F6 = F6 + 1
1740  IF F6 < 2 THEN 1760
1750  PRINT " ","CAREFUL TODAY ";F6;" CYCLES CROSS
       ZERO ****"
1760 F6 = 0
1770  NEXT X
1780  GOTO 3100
1790  GOTO 2200
1800  REM  * JULIAN DATE SUBR
1840 C9 = 0.001
1850 M9 = ( - 1) *  INT (((14 - M) / 12) + C9)
1860 J1 = D - 2447095 +  INT ((1461 * (Y1 + 4800 + M9)
/ 4) + C9)
```

```
1870 J2 = J1 +  INT ((367 * (M - 2 - 12 * M9) / 12) +
C9)
1880 J1 = J2 -  INT ((3 * (Y1 + 4900 + M9) / 400) + C9)
1890 J2 = J1
1900 W = J2 - 7 *  INT ((J1 / 7) + C9) + 1
1910 W =  INT (W + C9)
1920  RETURN
1930  REM  * CALENDAR SUBR
1960  REM
1970 J3 =  INT (J2 + C9)
1980 D8 = J2 - J3
1990 J2 = J3
2000 L = J2 + 2483589
2010 N =  INT ((4 * L / 146097) + C9)
2020 L = L -  INT (((146097 * N + 3) / 4) + C9)
2030 Y2 =  INT ((4000 * (L + 1) / 1461001) + C9)
2040 L = L -  INT (1461 * (Y2 / 4) + C9) + 31
2050 M1 =  INT (80 * (L / 2447) + C9)
2060 D1 = L -  INT ((2447 * M1 / 80) + C9)
2070  IF M1 > 0 THEN 2090
2080 D1 = D1 - 1
2090 L =  INT ((M1 / 11) + C9)
2100 D8 =  INT (10 * D8 + 0.5)
2110 D1 = D1 + (D8 / 10)
2120 M1 = M1 + 2 - 12 * L
2130 Y2 = 100 * (N - 49) + Y2 + L
2140 W = J2 - 7 *  INT ((J2 / 7) + C9) + 1
2150 W =  INT (W + C9)
2160  RETURN
2200  REM  * PLOTTING SUBR
2210 A1 = 7:H = J5:I = J6
2240 E1 =  - 1:F1 = 1:J = 1
2270 D =  - 1:E = 0:F = 1
2300 G = (F1 - E1) / 40
2310 G1 = G / 2
2320  GOSUB 310
2350  PRINT " "," THIS IS A BIORHYTHM CHART"
2360  PRINT " "
2370  PRINT " DAY"," "," ","        DATE"
2380  GOSUB 3020
2390  GOSUB 3080
2400 LN = 15
2410 X = H - J
2420  REM  *
2430 L1 = 0
2440 X = X + J
2450 J2 = X
2460  GOSUB 1960
```

```
2470 W1 = W
2480  IF X > I THEN 2970
2490  PRINT " ";D$(W1);"    ";
2500 D9 = 2 * (X - J4) * (3.1415927)
2510 A =  SIN (D9 / 23)
2520 B =  SIN (D9 / 28)
2530 C =  SIN (D9 / 33)
2540 Y = E1 - G
2545 Y = Y - 0.000000002
2550  PRINT " ";
2560  IF (Y + G) < 1 THEN 2590
2570  PRINT "    ";M$(M1);" ";D1;", ";Y2
2572 LN = LN + 1
2574  IF LN <  = 58 THEN 2582
2575  GOSUB 3080: GOSUB 3020
2576  PRINT  CHR$ (12): GOSUB 340
2578  GOSUB 3020: GOSUB 3080
2580 LN = 10
2582  GOTO 2420
2590 Y = Y + G
2600  IF Y > F1 THEN 2420
2610  IF  ABS (A - Y) < G1 THEN 2730
2620  IF A1 < 2 THEN 2550
2630  IF  ABS (B - Y) < G1 THEN 2770
2640  IF A1 < 3 THEN 2550
2650  IF  ABS (C - Y) < G1 THEN 2810
2660  IF A1 < 4 THEN 2550
2670  IF  ABS (D - Y) < G1 THEN 2850
2680  IF A1 < 5 THEN 2550
2690  IF  ABS (E - Y) < G1 THEN 2890
2700  IF A1 < 6 THEN 2550
2710  IF  ABS (F - Y) < G1 THEN 2930
2720  GOTO 2550
2730  PRINT "P";
2740 L1 = L1 + 1
2750  IF L1 = A1 THEN 2420
2760  GOTO 2560
2770  PRINT "E";
2780  GOTO 2740
2810  PRINT "M";
2820  GOTO 2740
2850  PRINT "-";
2860  GOTO 2740
2890  PRINT "I";
2900  GOTO 2740
2930  PRINT "+";
2940  GOTO 2740
2970  REM  *
2980  GOSUB 3080
2990  GOSUB 3020
```

```
3000  PRINT
3010  GOTO 3100
3020  REM  * PRINT SUBR
3040  PRINT  TAB( 10)E1,"          CRITICAL
+";F1
3050  PRINT " ","          PERIOD"
3060  PRINT " "
3070  RETURN
3080  PRINT "
:....:....:....:....:....:....:....:....:....:"
3090  RETURN
3100  PRINT  CHR$ (12)
3102  PRINT  CHR$ (9);"40N"
3104  PR# 0
3110  PRINT
3120  INPUT "WANT DIFFERENT DATES(Y/N)? ";ANS$
3125  IF  LEFT$ (ANS$,1) = "Y" OR  LEFT$ (ANS$,1) = "N"
THEN 3140
3130  PRINT "ENTRY INCORRECT....REENTER!": FOR I = 1
       TO 1500: NEXT I: GOTO 3110
3140  IF  LEFT$ (ANS$,1) = "Y" THEN 1080
3150  PRINT : INPUT "WANT BIORHYTHM FOR ANOTHER
       PERSON(Y/N)? ";ANS$
3160  IF  LEFT$ (ANS$,1) = "Y" OR  LEFT$ (ANS$,1) = "N"
THEN 3180
3170  PRINT : PRINT "ENTRY INCORRECT....TRY AGAIN!":
       FOR I = 1 TO 1500: NEXT I: GOTO 3150
3180  IF  LEFT$ (ANS$,1) = "Y" THEN 50
3230  END
```

圖11-17 Biorhythm 程式

11.3.6 日 曆(CALENDAR)

```
1   REM *** CALENDAR ***
2   REM *
3   REM * WRITTEN BY:JL CAMPBELL
4   REM *        DATED:07/24/80
5   REM *
6   REM * COMPUTER:APPLE II
7   REM * LANGUAGE:APPLESOFT II
8   REM *
9   REM **********************
10   TEXT : CLEAR : POKE 216,0:A =  FRE (0)
20   DIM DM(12)
22 WD$ = "SMTWTFS":Q$ = "1ST2ND3RD4TH"
30   HOME : PRINT : PRINT  TAB( 12)"*** CALENDAR
      ***": PRINT
40   FOR I = 1 TO 12: READ DM(I): NEXT I
42   DATA  31,28,31,30,31,30,31,31,30,31,30,31
50   INPUT "START WHICH YEAR? = ";YR
55   IF YR < 100 THEN YR = YR + 1900
60 Y = YR - 1:M = 13:D = 1
65 N = D + 2 * M +  INT (0.6 + (M + 1)) + Y
70 N = N +  INT (Y / 4) -  INT (Y / 100) +  INT (Y
     / 400) + 2
75 N =  INT ((N / 7 -  INT (N / 7)) * 7 + 0.5)
80 DY = N
140   IF  INT (YR / 4) = YR / 4 THEN DM(2) = DM(2) + 1
150   VTAB (4): CALL  - 958
160   PRINT "DO THE FOLLOWING;"
170   PRINT  TAB( 5)"1- LOAD PRINTER PAPER"
180   PRINT  TAB( 5)"2- TURN ON PRINTER"
190   PRINT  TAB( 5)"3- SET TOP OF FORM"
200   PRINT : INPUT "TYPE RETURN KEY TO CONTINUE... ";A$
210   PRINT : PRINT "PRINTER TYPE;"
220   PRINT  TAB( 5)"1- SERIAL"
230   PRINT  TAB( 5)"2- PARALLEL"
250   PRINT : INPUT "WHICH ONE? ==> ";SP
280   PRINT : INPUT "WHICH SLOT?(1-7) = ";SL
290   PR# SL
300   IF SP = 2 THEN  PRINT  CHR$ (9);"80N"
305 IX = DY
310   FOR I = 1 TO 10 STEP 3
312   PRINT  TAB( 29)"*** ";YR;"  CALENDAR ***"
316   PRINT  MID$ (Q$,I,3);" QUARTER"
```

```
320  FOR J = I TO I + 2
330  FOR K = 1 TO 34: PRINT "=";: NEXT K
350  ON J GOTO 360,370,380,390,400,410,420
      ,430,440,450,460,470
360  PRINT "  JANUARY  ";: GOTO 500
370  PRINT " FEBRUARY ";: GOTO 500
380  PRINT "   MARCH   ";: GOTO 500
390  PRINT "   APRIL   ";: GOTO 500
400  PRINT "    MAY    ";: GOTO 500
410  PRINT "    JUNE   ";: GOTO 500
420  PRINT "    JULY   ";: GOTO 500
430  PRINT "   AUGUST  ";: GOTO 500
440  PRINT "SEPTEMBER ";: GOTO 500
450  PRINT " OCTOBER  ";: GOTO 500
460  PRINT " NOVEMBER ";: GOTO 500
470  PRINT " DECEMBER ";: GOTO 500
500  FOR K = 1 TO 34: PRINT "=";: NEXT K: PRINT
510  PRINT " "
515  PRINT YR;
520  FOR K = 1 TO 7: POKE 36,10 * (K - 1) + 8:
      PRINT  MID$ (WD$,K,1);: NEXT K: PRINT " "
530  PRINT " "
540  FOR K = 1 TO 78: PRINT "*";: NEXT K: PRINT
550  PRINT " "
600  FOR K = 1 TO DM(J)
610  POKE 36,10 * IX + 8
620  PRINT K;
630 IX = IX + 1
640  IF IX <  > 7 THEN 660
650  PRINT " ": PRINT " ":IX = 0
660  NEXT K
670  PRINT " ": PRINT " "
675  PRINT " "
680  NEXT J
690  FOR K = 1 TO 78: PRINT "=";: NEXT K: PRINT " "
700  PRINT  CHR$ (12)
720  NEXT I
2000  IF SP = 2 THEN  PRINT  CHR$ (9);"40N"
2010  PR# 0
2020  HOME : VTAB (10): PRINT "GOODBYE FOR NOW...": END
```

圖11-18 Calendar程式

11.3.7 唱歌的程式

這個程式將使用低解析度圖形在演奏 Aulde Lang Syne 音樂的同時將一個圖形顯示在螢光幕上。這個樂曲是由行號 15 到 20 的敍述建立出來的。

```
   1 REM **** AULDE LANG SYNE ****
   2 REM *
   3 REM * WRITTEN BY:JL CAMPBELL
   4 REM *      DATED:05/30/78
   5 REM *
   6 REM * COMPUTER:APPLE II
   7 REM * LANGUAGE:APPLE BASIC
   8 REM *
   9 REM ************************
  10 TEXT: CLEAR: POKE 216,0: A = FRE(0)
  11 DIM F(12)
  15 POKE 2,173:POKE 3,48:POKE 4,192:POKE 5,136
:POKE 6,208:POKE 7,4:POKE 8,198:POKE 9,1:POKE
10,240
  20 POKE 11,8:POKE 12,202:POKE 13,208:POKE
14,246:POKE 15,166:POKE 16,0:POKE 17,76:POKE
18,2:POKE 19,0:POKE 20,96
  30 F(1)=138:F(2)=126:F(3)=112:F(4)=100
  32 F(5)=25:F(6)=85:F(7)=75:F(8)=70
  34 F(9)=64:F(10)=57:F(11)=52:F(12)=48
  40 CALL -936:GR:COLOR=7
  50 VTAB 22: TAB 10: PRINT "** AULDE LANG SYNE
**":
     VTAB 23: PRINT "*********": VTAB 23:
     TAB 30:PRINT "*********"
  90 FOR N = 1 TO 2:IDX = 0
 100 P=F(2):D=200:GOSUB 1000:GOSUB 500
 110 P=F(5):GOSUB 1000:GOSUB 500
 120 P=F(4):D=100:GOSUB 1000:GOSUB 500
 130 P=F(5):D=200:GOSUB 1000:GOSUB 500
 140 P=F(7):GOSUB 1000:GOSUB 500
 150 P=F(6):D=250:GOSUB 1000:GOSUB 500
 160 P=F(5):D=100:GOSUB 1000:GOSUB 500
 170 P=F(6):D=200:GOSUB 1000:GOSUB 500
 180 P=F(7):GOSUB 1000:GOSUB 500
 190 P=F(5):D=250:GOSUB 1000:GOSUB 500
 195 D=100:GOSUB 1000: GOSUB 500
 200 P=F(7):D=200:GOSUB 1000:GOSUB 500
 205 P=F(9):GOSUB 1000:GOSUB 500
```

```
 210 P=F(10):GOSUB 1000:GOSUB 1000:GOSUB 1000:GOSUB
500:GOSUB 1000:GOSUB 500
 215 P=F(9):GOSUB 1000:GOSUB 500
 220 P=F(7):D=100:GOSUB 1000:GOSUB 500
 225 D=200:GOSUB 1000:GOSUB 500
 230 P=F(5):GOSUB 1000:GOSUB 500
 235 P=F(6):D=100:GOSUB 1000:GOSUB 1000:GOSUB
1000:GOSUB 500
 240 P=F(5):GOSUB 1000:GOSUB 500
 245 P=F(6):D=200:GOSUB 1000:GOSUB 500
 250 P=F(7):GOSUB 1000:GOSUB 500
 255 P=F(5):D=100:GOSUB 1000:GOSUB 1000:GOSUB
1000:GOSUB 500
 260 P=F(3):GOSUB 1000:GOSUB 500
 265 D=200:GOSUB 1000:GOSUB 500
 270 P=F(2):GOSUB 1000:GOSUB 500
 275 P=F(5):GOSUB 1000:GOSUB1000:GOSUB  1000:GOSUB
500
 280 P=F(10):GOSUB 1000:GOSUB 500
 285 P=F(9):D=100:GOSUB 1000:GOSUB 1000:GOSUB 1000:
GOSUB 500
 290 P=F(7):D=100:GOSUB 1000:GOSUB 500
 295 D=200:GOSUB 1000:GOSUB 500
 300 P=F(5):GOSUB 1000:GOSUB 500
 305 P=F(6):D=100:GOSUB 1000:GOSUB1000:GOSUB
1000:GOSUB 500
 310 P=F(5):GOSUB 1000:GOSUB 500
 315 P=F(6):D=200:GOSUB 1000:GOSUB 500
 320 P=F(10):GOSUB 1000:GOSUB 500
 325 P=F(9):D=100:GOSUB 1000:GOSUB 1000:GOSUB
1000:GOSUB 500
 330 P=F(7):GOSUB 1000:GOSUB 500
 335 D=200:GOSUB 1000:GOSUB 500
 340 P=F(9):GOSUB 1000:GOSUB 500
 345 P=F(10):GOSUB 1000:GOSUB 1000:GOSUB 1000:GOSUB
500:GOSUB 1000:GOSUB 500
 350 P=F(9):D=100:GOSUB 1000:GOSUB 1000:GOSUB
1000:GOSUB 500
 355 P=F(7): GOSUB 1000:GOSUB 500
 360 D=200:GOSUB 1000:GOSUB 500
 365 P=F(5):GOSUB 1000:GOSUB 500
 370 P=F(6):D=100:GOSUB 1000:GOSUB 1000:GOSUB
1000:GOSUB 500
 375 P=F(5):GOSUB 1000:GOSUB 500
 380 P=F(6):D=200:GOSUB 1000:GOSUB 500
 385 P=F(7):GOSUB 1000:GOSUB 500
 390 P=F(5):D=100:GOSUB 1000:GOSUB 1000:GOSUB
1000:GOSUB 500
```

```
 400 P=F(3):GOSUB 1000:GOSUB 500
 405 D=200:GOSUB 1000:GOSUB 500
 410 P=F(2):GOSUB 1000:GOSUB 500
 415 P=F(5):GOSUB 1000:GOSUB 1000:GOSUB 1000:GOSUB
500
 420 FOR K = 1 TO 250:NEXT
 430 COLOR = 1:PLOT 7,11:PLOT 4,12:PLOT 2,13:PLOT
3,9:PLOT 6,4: PLOT 7,28:PLOT 5,24:PLOT 4,20
 440 PLOT 20,13:PLOT 36,12:PLOT 27,9:PLOT 22,15
 450 COLOR = 13:GOSUB 2532
 460 COLOR = 15:PLOT 10,17:PLOT 6,19:PLOT 4,18:PLOT
8,12:PLOT 3,6
 480 NEXT N
 490 END
 497 REM *
 498 REM ** DELAY & DRIVE PICTURE
 499 REM *
 500 FOR J = 1 TO 50:NEXT J
 510 GOSUB 2550-IDX
 520 RETURN
 997 REM *
 998 REM * PRODUCE TONE
 999 REM *
1000 POKE 0,P:POKE 1,D:CALL 2
1010 RETURN
1997 REM *
1998 REM ** DRAW GRAPHICS PICTURE
1999 REM *
2000 COLOR = 15:PLOT 8,17
2010 PLOT  6,18:PLOT 5,12
2020 PLOT  5,14:PLOT 6,10:PLOT 4,6
2030 PLOT  4,15:PLOT 2,11:PLOT 0,5
2040 PLOT  2,12:PLOT 3,8 :PLOT 0,0
2050 PLOT  5,12
2060 PLOT  8,14
2070 COLOR = 11:HLIN 37,38 AT 10
2080 HLIN 34,38 AT 11
2090 PLOT 23,10:PLOT 32,9
2100 PLOT 27,9
2110 PLOT 31,10
2120 COLOR = 11:PLOT 10,17
2130 HLIN 9,11 AT 18
2140 HLIN 3,4  AT 18
2150 PLOT 3,20
2160 PLOT 3,19
2170 PLOT 11,15
2180 PLOT 8,14
```

```
2190 COLOR = 11:PLOT 10,16
2200 PLOT 11,19:VLIN 29,38 T 7
2210 FOR J = 0 TO 8:HLIN 3,11 AT 21 + J:NEXT J
2220 HLIN 4,11 AT 20
2230 HLIN 4,10 AT 19
2240 HLIN 5,8 AT 18
2250 HLIN 5,8 AT 17
2260 HLIN 6,9 AT 16
2270 HLIN 8,10 AT 15
2280 PLOT 11,12
2290 PLOT 10,13:PLOT 10,12
2300 HLIN 9,10 AT 14
2310 COLOR = 11:HLIN 11,38 AT 12
2320 HLIN 15,38 AT 13
2330 HLIN 17,38 AT 14
2340 HLIN 19,38 AT 15:PLOT 21,12:PLOT 30,12
2350 HLIN 21,38 AT 16:PLOT 25,13:PLOT 33,14
2360 COLOR = 15:HLIN 11,14 AT 11
2370 HLIN 11,14 AT 13
2380 HLIN 14,16 AT 10
2390 HLIN 14,16 AT 14
2400 HLIN 16,18 AT 9
2410 HLIN 16,18 AT 15
2420 HLIN 18,20 AT 8
2430 HLIN 18,20 AT 16
2440 COLOR = 15:VLIN 8,16 AT 39
2450 HLIN 20,39 AT 7
2460 HLIN 20,39 AT 17
2470 VLIN 15,30 AT 12
2480 VLIN 15,30 AT 2
2490 HLIN 2,6 AT 30:HLIN 8,12 AT 30
2500 VLIN 31,38 AT 8
2510 VLIN 31,38 AT 6
2520 HLIN 2,12  AT 39
2530 COLOR = RND (15)+1
2532 PLOT 37,33:HLIN 36,39 AT 39:VLIN 33,39 AT 38
2540 VLIN 33,39 AT 31:HLIN 32,33 AT 33:VLIN 33,39
AT 34:HLIN 32,33 AT 36:HLIN 32,33 AT 39
2550 VLIN 34,35 AT 26:HLIN 26,26 AT 36:HLIN 26,28
AT 33:VLIN 33,39 AT 29
2560 PLOT 22,33:HLIN 21,24 AT 39:VLIN 33,39 AT 23
2570 IDX = IDX + 10
2580 VLIN 24,30 AT 15:VLIN 24,30 AT 18:HLIN 16,17
AT 27
2590 VLIN 25,30 AT 20:VLIN 25,30 AT 23:HLIN 21,22
AT 24:HLIN 21,22 AT 27
2600 VLIN 24,30 AT 25:VLIN 24,27 AT 28:HLIN 26,27
```

```
AT 24:HLIN 26,27 AT 27
2610 VLIN 24,30 AT 30:VLIN 24,27 AT 33:HLIN 31,32
AT 24:HLIN 31,32 AT 27
2620 VLIN 24,25 AT 35:VLIN 24,25 AT 39:PLOT
36,26:PLOT 38,26:VLIN 27,30 AT 37
2630 RETURN
```

圖11-19　Auld Lang Syne程式

11.4 總　結

對於本章中所介紹的程式不可能每一個讀者都能夠完全地喜愛它們，但是我們希望您能夠對其中的一部份感到有趣。有一些程式可以爲您做一些工作，而另外的一些則供做娛樂及練習用。

本章中提供的遊戲程式並不是只提供娛樂罷了，它們在指令方面的使用情形要比娛樂方面的功用大得多了；我們希望您也能夠自己設計您的遊戲程式。

當我們想要做英制及公制的轉換工作的時候，您就可以使用轉換的程式達成目的了。

最後對於數學方面有興趣的讀者也可以在本章中找到像是高斯消去法等等有用的程式，我們並不希望您在看完了本章之後就能夠設計出很好的程式來，我們只是希望您能夠對程式中使用的指令都能瞭解，並且能夠在以後其他的問題中設計您自己的程式。

您也許注意到了有一些程式是在許久以前就設計好了的，它們並沒有完全依照程式架構的原則設計，這一點希望您能原諒。

第12章

滙 總

12.0 大 綱

您現在應該已經具備了在 Apple][上設計一個完整而有效的結構式程式的能力了，在這一章中我們將告訴您如何將所有您學會的技巧結合在一起完成這個工作。

我們將使用一個目前在市場上銷售的程式系統做爲解說的例子；當我們介紹程式中的每一個主要部份的時候，我們也會告訴您這是使用書中某一段中陳述的方法，而您也就可以參考那一段的說明了。

請注意這一章討論的方法是依照第二章中所陳述的處理過程安排討論程序的。

12.1 選擇問題

當我們設計一個投資決策(INVESTMENT DECISIONS)的系統時，我們必須考慮幾個因素。第一，所有的問題必須密切的相關聯，也就是說它們必須具有重複發生的特性，那麼它們才可以使用 Apple][來解決。其次，解決的方法必須能夠適用於同樣性質的許多問題，這樣子才能夠在市場上銷售。

現在觀察這個投資決策的程式系統看看它是否與下面的需要符合：

1. 貸款分期定額償還表（Loan schedules ）
2. 定期存款到期本利和表（ Savings schedules ）
3. 年金終值表（ Annuity schedules ）
4. 折舊表（ Depreciation schedules ）
5. 攤銷表（ Ammortization schedules ）
6. APR 表（APR schedules ）
7. 償還點計算法（ Payback method ）
8. 淨現值法（Net Present Value (NPV)）
9. 投資報酬率（Internal Rate of Return (IRR) ）
10. 獲利率指標（ Profitability Index (PI) ）

您可以看出來上面的每一個程式都是與財務計劃(financial planning)或是資本運用（capital budgeting ）脫離不了關係的。對於某一個使用者來說他（或她）也許並不需要使用每一個程式，但是至少有一個或更多的程式可以幫他（或她）做事。由於這些程式之間都互相關聯著，因此對許多的使用者而言，他們可能都要使用所有的程式。

每一個程式都具有一個特性，那就是它們可以解決任何合法的輸入資料，並且產生出正確的結果。例如，我們使用貸款分期定額償還表（loan sc-hedule）程式得到的結果，會因爲我們輸入的 "NOTE AMMOUNT BOR-ROWED=" 值的不同而不同；因此，除非一個人在同樣的利率下借了相同的錢，並且按照相同的付款方式還錢，否則程式執行的結果必定不相同。

這些程式同時也解決了一些經過明確定義的問題，而產生的結果都是很適合 Apple][的表達能力的。

請對程式的特性注意兩點事情，第一就是程式都是可以反覆執行的，其次它們都是適合在 Apple][上執行的。

12.2 對問題的描述

我們的目的是要建立一個商業上使用的程式系統，全部存在一片磁片上

，並且能夠解決在 12.1 節中描述的所有項目。

這個程式系統我們將使用由上往下的方式設計出來，下面就是它的格式。

```
                HELLO
                  !
                ID000
                  !
  +---+---+---+---+---+---+---+---+---+
  !   !   !   !   !   !   !   !   !   !
ID010 ! ID020 ! ID030 ! ID040 ! ID050 !
      !       !       !       !       !
    ID060   ID070   ID080   ID090   ID100
```

就如同您在這個程式結構上所見到的一般， HELLO 程式就是帶領您進入整個程式系統的程式； HELLO 程式會自動地執行 ID.000 程式，ID.000 也就是整個程式系統的執行程式；當您在 ID.000 內時，您就可以選擇您要執行的程式，等到這個程式被執行完畢之後，您又回到 ID.000 程式內了。

這個問題並不需要任何的文字資料檔，我們只需要建立一個Applesoft][的程式檔就可以了。

在這個程式系統中所有的程式都依照由上往下的方式組織而成；每一個程式都需要以下的組織元素：

* 一所有的輸入
* 一處理的過程
* 一所有的輸出

每一個程式都依照上面的次序組織而成。

至於螢幕顯示資料的格式則依照下面的原則決定：

* 一螢幕的顯示情況必須令人愉悅。
* 一需要讓出空間的時候，螢幕是往上方捲動的。
* 一錯誤訊息將顯示在螢幕底端行的中間。
* 一螢幕上的選擇是由使用者決定的。

處理錯誤的程式段將包含下面幾項，但並不僅限於這些：

* 一做手册輸入的範圍測試。
* 一檢查 Yes/No的輸入。
* 一檢查資料是否有效。
* 一當需要的時候將檢查輸入的是否為負數值。
* 一鎖住鍵盤的動作將被使用。
* 一程式中斷執行的情況將被找出。

所有的計算工作都是依照已知的公式計算的，而輸入的值都將被測試是否為有效的資料。

我們不論在存款或是貸款時利息的計算方法大都是使用複利(Compound interest)的方式。假設目前存款$1000,00 ，利息計算方法是三個月結算一次（使用複利的方式），那麼在第1年之後您的帳目將如下面的表一般。

#	本 金	利 息	總 數
1	$1,000.00	$12.50	$1,012.50
2	1,012.50	12.66	1,025.16
3	1,025.16	12.81	1,037.97
4	1,037.97	12.97	1,050.94

您可以由這個表中看出來，您帳戶內的錢每一季（三個月）累計一次，而新一季的錢數就是使用上一季的錢數加上利息而得來的；這就是複利的計算方法。至於程式中所使用的計算公式您可以參考下面的討論，詳細的情形請閱讀關於財務管理、簿記方面的書籍。

數學公式：(Mathematics):

在接下去的部份，我們將討論這個程式系統中計算所依據的公式。

貸款分期定額償還表 (Loan Schedules) :

MP =一定本金定期定額償還每期應償金額 (Regular Payment)
i ＝年利率 (Interest Rate, Annual)
c =計算複利的單位時間 (Compounding Periods)
Y =年數 (Years)
M =月數 (Months)
PV =本金 (Principal Amount)

```
Y   = (Y * 12 + M)/12

         PV * (1/c)
MP  = ---------------------
      (1 - (1 + i/c)^(-Y*c)
```

定期存款到期本利和表 (Savings Schedules) :

FV =複利終值 (Future value)
PV =現值 (Present Value)
i =複利 (Interest)
c =計算複利的單位時間 (Compounding Periods)
Y =年數 (Years)
M =月數 (Months)

```
 Y = (Y * 12 + M)/12
FV = PV * (1 + i/c)^(Y*c)
```

年金終值表 (Annuity Schedules) :

PV =定期付款額 (Payment Value)

FV =複利本利和 (Future Value)

i =複利 (Interest)

c =每年付款的次數 (Payments/year)

Y =年數 (Years)

```
FV = PV * (1 + i/c) * ((1 + i/c)^(Y*c) - 1)/(i/c)
```

注意：每一個變數都可以按照上面的公式解得它的值。

折 舊 (Depreciation)：

YD =年度折舊 (Yearly Depreciation)

PP =買價 (Purchase Price)

SV =殘值 (Salvage Value)

YR =耐用年限 (Useful Years)

HR =耐用時數 (Useful Hours)

注意：如果 SV <= 10% 則表示已無殘餘價值存在。

直線法 (Straight Line)：

```
        (PP - SV)
YD =    -------
          YR
```

＊倍數餘額遞減法 (Double Declining Balance)：

```
        (PP - SV)   2.0
YD =    ------- *   ---
          YR        YR
```

＊注意：此公式依照正常情況應該是 $YD = \frac{(PP-SV)}{YR} * 2.0$ ，供讀者參考。

年數合計法 (Sum-of-Digits):

SUM = 由 1 到 YR 的年數合計值。

i = 某一特定的年數。

```
          (YR + 1)-i
YD = ----------- * PP
             SUM
```

生產數量法 (Units-of-Production):

```
          (PP - SV)
YD =  -------
             HR
```

攤銷表 (Amortization Schedules):

注意：依據貸款分期定額償還表（Loan Schedules）公式計算。

APR 表 (APR Schedules):

注意：依據貸款分期定額償還表（Loan Schedules）公式計算。

投資指數 (Investment Indices):

償還點計算法 (Payback Method):

這個公式計算出初期投資－流入現金總和＝ 0 的那一點。

淨現值法 (Net Present Value):

NPV ＝淨現值（ Net Present Value ）

Rn ＝第 n 次期間流入的現金（ Cash Flow-period n ）

i ＝投資現值（ Initial Investment ）

C ＝折扣率（利率）（ Discount Rate(interest) ）

$$NPV = \sum_{j=1}^{n} \frac{Rj}{(1+i)^j} - C$$

投資報酬率　（ Internal Rate of Return ）：

$$IRR = \sum_{j=1}^{n} \frac{Rj}{(1+i)^j} = C$$

注意：i 精確至一個百分點的千分之一，至於 IRR 的值與投資現值的誤差在 $10.00 之內。

獲利率指標　（ Profitability Index ）：

PI ＝獲利率指標（ Profitability Index ）

$$PI = \frac{\sum_{j=1}^{n} \frac{Rj}{(1+i)^j}}{C}$$

注意：PI ＝ 1.0 表示位於損益兩平點（ breakeven point ）。

PI ＞ 1.0 表示有獲利的可能。PI ＜ 1.0 表示無利可圖。

整個程式系統中程式的測試工作將在下列的時間完成：

* 一當這個程式系統完成的時候
  一檢查語法是否正確
  一檢查程式間的連接關係
  一檢查初步的偵錯工作

* 一當每個程式的主程式部份完成後
  一檢查語法
  一檢查手册中選擇的工作
  一檢查與主要副程式間的連接情形
  一檢查副程式返回主程式的情形

* 一當主要的副程式完成了之後
  一檢查語法
  一檢查處理的過程
  一檢查與中型副程式間的連接
  一檢查返回主程式的情形
  一檢查初步的偵錯工作

* 一當所有程式行完成之後
  一檢查語法
  一檢查所有的連接情形
  一使用資料測試所有的處理情形
  一檢查所有偵錯的情形
  一檢查程式的完整性
  一檢查程式的正確性

* 一當要推出供應市場時
  一採取 Alpha 測試觀點

一採取 Beta 測試觀點

＊一在推出供應市場之前再做最後的測試與修正的工作。

12.3 設計解決的程式

在我們詳細地描述過了問題之後，現在來設計解決的程式也比較簡單了。由於我們已經知道了問題需要的所有要素及計算的公式，現在我們所要做的只是把它們滙編在一起成爲一個完整的體系。對於這個結構式系統的描述，我們將依照圖 12 － 1 的流程圖來做說明。

在這個系統中第一個程式就是 HELLO 程式，我們希望它能在磁片 boot 的時候就自動地執行；爲了要達成這個目的，我們必須在設定磁片格式的同時將這個程式存到磁片內。

在 HELLO 程式執行的時候，它就會呼叫名字叫做 ID.0000 的執行程式；這個程式就是系統程式中的手册程式，藉著它我們可以進入程式系統中的任何一個程式。依照我們結構化程式的標準，這個程式將呼叫其他的附屬程式，而這些附屬程式都會在手册中顯現出來供您選擇。在您做了一個選擇之後，這個程式就會檢查您的輸入是否合法，然後就使用您輸入的資料及 CHR$(4) 磁碟命令呼叫您選擇的附屬程式了。

這個軟體系統的其他部份就是一個一個的程式；我們可以使用在 12.2 節中討論的數學公式做爲設計程式的主體，再加入其他的部份就可以了；我們也會把在第七章中介紹的像是處理數字及"print using"等等的副程式加入，這些副程式使得我們能夠在輸出的時候得到整齊漂亮的數字，並且可以幫助我們找出錯誤的資料。當我們在設計程式的時候，我們必須記得前面所討論的種種需求，並把它們考慮在我們的程式中。

程式系統中的每一個程式都是獨立的；那就是說，任何一個程式的輸入、輸出與處理的過程都不會與其他的程式產生關聯（除了被執行程式呼叫之外）。在某一個程式執行完畢之後，執行就會回到 ID.0000 程式內去。我們現在把設計這個程式系統時的幾點主要考慮敍述如下：

* 一每一個程式都是獨立的。
* 一每一個程式都被執行程式（ ID.0000 ）呼叫而執行，並且在執行完畢之後再回到執行程式去。
* 一所有的資料都由鍵盤輸入。
* 一所有的輸出將顯示在 CRT 上。
* 一印報表的工作是可有可無的，但是攤銷表（Amortization Schedule）是一個例外。
* 一對於貸款分期定額償還表（ Loan Schedule ）及複利終值表（ Savings Schedule ）兩個程式都沒有報表可以印出。
* 一我們將依照前面描述的次序將程式設計完成。

ID.000 = 執行程式

```
0   POKE 1011,224
1   REM ***** ID.000 ***********
2   REM *
3   REM * WRITTEN BY:JL CAMPBELL
4   REM *      DATED:12/26/80
5   REM *
6   REM * LANGUAGE:FPBASIC
7   REM * COMPUTER:APPLE II
8   REM * INVESTMENT DECISIONS
9   REM ************************
10   TEXT : CLEAR : POKE 216,0: HOME
20   DIM A$(10)
40   ONERR  GOTO 10000
50   GOTO 900
51   REM **************
52   REM * GEN'L SUBR *
53   REM **************
55   INPUT "WHICH ONE? ==> ";N: RETURN
60   VTAB (23): HTAB 13: INVERSE : PRINT "ILLEGAL
ENTRY";: NORMAL : PRINT  CHR$ (7)
64   FOR L = 1 TO 1000: NEXT L: RETURN
900   FOR I = 1 TO 10: READ A$(I): NEXT I
902   DATA  1 -LOAN SCHEDULES,2 -SAVINGS SCHEDULES,3
-ANNUITY SCHEDULES,4 -DEPRECIATION SCHEDULES,5
-AMORTIZATION SCHEDULES,6 -APR SCHEDULES(EG AUTOS),7
-PAYBACK METHOD,8 -NPV=NET PRESENT VALUE
904   DATA  9 -IRR=INTERNAL RATE OF
RETURN,10-PROFITABILITY INDEX
```

圖12-1 Hello程式

12.4 程 式

下面就是投資決策的所有程式。

HELLO = Boot 程式

```
0   POKE 1011,224
1   REM ***** HELLO **********
2   REM *
3   REM * -- DOS 3.3:HELLO
4   REM *       DATED:08/25/80
5   REM *
6   REM * LANGUAGE:FPBASIC
7   REM * COMPUTER:APPLE II
8   REM *
9   REM **********************
10   TEXT : CLEAR : POKE 216,0: HOME
15   FOR I = 1 TO 39: PRINT "*";: NEXT I: PRINT
20   FOR I = 1 TO 20: PRINT "*"; TAB( 39)"*": NEXT I
25   FOR I = 1 TO 39: PRINT "*";: NEXT I: PRINT
30   PRINT
32   POKE 32,2: POKE 33,36: POKE 34,2: POKE 35,19
34   HOME : PRINT : PRINT  TAB( 11)"MESA  RESEARCH":
PRINT
36   PRINT  TAB( 8)"INVESTMENT DECISIONS": PRINT
38   PRINT  TAB( 17)"BY": PRINT
40   PRINT  TAB( 11)"J. L. CAMPBELL"
42   VTAB 14: PRINT  TAB( 12)"VERSION  1.7"
44   VTAB (16): PRINT  TAB( 11)"DATED:01/22/81"
46   VTAB (18): PRINT  TAB( 11)"COPYRIGHT 1981"
50   FOR L = 1 TO 2500: NEXT L
60   POKE 32,0: POKE 33,40: POKE 34,0: POKE 35,24
70   HOME
90 P$ = "ID.000"
300   PRINT  CHR$ (4);"RUN ";P$;",D1"
```

```
1000  HOME : PRINT  TAB( 13)"MESA  RESEARCH"
1002  PRINT  TAB( 10)"INVESTMENT DECISIONS": PRINT
1005  PRINT "MAIN MENU:"
1010  FOR I = 1 TO 10: PRINT  TAB( 10)A$(I): NEXT I
1028  PRINT
1030  PRINT  TAB( 10)"0 -EXIT SYSTEM": PRINT
1040  VTAB (18): CALL  - 958: GOSUB 55
1070  IF  INT (N) = N THEN 1090
1072  GOSUB 60: GOTO 1040
1090  IF N = 0 THEN 1200
1095  IF N > 0 AND N < 11 THEN 1110
1100  GOTO 1072
1110  VTAB (4 + N)
1115  INVERSE : PRINT "REQUEST: ";A$(N): NORMAL
1125  VTAB (20): CALL  - 958: INPUT "IS THAT
CORRECT?(Y/N) ";Q$
1130  IF  LEFT$ (Q$,1) = "Y" THEN 1140
1132  IF  LEFT$ (Q$,1) = "N" THEN 1000
1134  GOSUB 60: GOTO 1125
1140 N =  INT (N * 10 + 0.5) + 10000
1150 P$ = "ID." +  RIGHT$ ( STR$ (N),3)
1160  PRINT  CHR$ (4);"RUN ";P$
1200  HOME : VTAB (10): PRINT "GOODBYE FOR NOW...."
1210  CLEAR : NEW : END
10000 E =  PEEK (222): VTAB (22): CALL  - 958
10005  PRINT "ERROR = <";E;">";
10010  IF E = 255 THEN  PRINT " IS CRTL-C ": POKE
222,0: GOTO 11000
10020  IF E = 254 THEN  PRINT " IS DATA ERROR": CLEAR
RESTORE : GOSUB 64: GOTO 900
10030  GOTO 1200
11000  VTAB (23): CALL  - 958: INPUT "DO YOU WISH TO
QUIT?(Y/N) = ";N$
11010  IF  LEFT$ (N$,1) = "Y" THEN 1200
11020  IF  LEFT$ (N$,1) = "N" THEN 1040
11030  GOSUB 60: GOTO 11000
```

圖12-2　執行程式

ID.010 =貸款分期定額償還表程式

```
0   POKE 1011,224
1   REM ***** ID.010 ***********
2   REM *
3   REM * WRITTEN BY:JL CAMPBELL
4   REM *       DATED:05/24/78
5   REM *
6   REM * COMPUTER:APPLE II
7   REM * LANGUAGE:APPLESOFT II
8   REM * LOAN SCHEDULES
9   REM ******************
10   TEXT : CLEAR : POKE 216,0:A =  FRE (0)
12   DIM PD(5),CH$(4)
20   FOR I = 1 TO 5: READ PD(I): NEXT I
22   DATA  1,2,4,12,365
30   FOR I = 1 TO 4: READ CH$(I): NEXT I
32   DATA  1- REGULAR PAYMENT ON NOTE,2- PRESENT VALUE
ON NOTE,3- # OF REGULAR PAYMENTS,4- ANNUAL INTEREST
RATE
50   GOTO 900
51   REM **************
52   REM * GEN'L SUBR *
53   REM **************
55   INPUT "WHICH ONE? ==> ";N: RETURN
60   VTAB (22): HTAB 10: INVERSE : PRINT "ILLEGAL
ENTRY": NORMAL : PRINT  CHR$ (7)
65   FOR L = 1 TO 1000: NEXT L: RETURN
100   VTAB (23): INPUT "TYPE RETURN KEY TO CONTINUE...
";A$
105   VTAB (10): CALL  - 958: RETURN
499   REM * NUMBER FILTER
500 IX = 1:B$ = "":SW = 0
505   VTAB (V): HTAB H + IX
510   PRINT "+";
515   VTAB (V): HTAB H + IX
520   GET X$
525   IF  ASC (X$) = 13 THEN 580
530   IF  ASC (X$) = 8 THEN 560
532   IF IX > 1 THEN 540
535   IF  ASC (X$) = 45 THEN 550
540   IF  ASC (X$) = 46 AND SW = 0 THEN SW = 1: GOTO 550
545   IF  ASC (X$) < 48 OR  ASC (X$) > 57 THEN 515
550 B$ = B$ + X$: PRINT X$
555 IX = IX + 1: GOTO 505
560   IF IX = 1 THEN 505
565   HTAB H + IX: PRINT " ";: HTAB H + IX - 1
```

```
567  IF  MID$ (B$, LEN (B$),1) = "." THEN SW = 0
570 B$ =  MID$ (B$,1, LEN (B$) - 1)
575 IX = IX - 1: GOTO 505
580  CALL  - 868
585  VTAB (V): HTAB H + 1: PRINT B$
590  RETURN
698  REM * SUMMARY DATA
700  PRINT "TOTAL # OF PAYMENTS = "; INT (Y * C + 0.5):
RETURN
705  PRINT "REGULAR PAYMENT     =$";MP: RETURN
710  PRINT "THE PRINCIPAL AMOUNT=$";PV: RETURN
715  PRINT "THE ANNUAL INTEREST = ";I * 100;"%": RETURN

720  PRINT "COMPOUNDING PERIODS = ";C: RETURN
799  REM * INPUT DATA
800  VTAB (4): IF S = 4 THEN 820
805  PRINT "ANNUAL INTEREST RATE = ";
806 V = 4:H = 23: GOSUB 500:I =  VAL (B$)
810 I = I / 100
820  VTAB (5): IF S = 1 THEN 830
825  PRINT "REGULAR PAYMENT      =$";
826 V = 5:H = 23: GOSUB 500:MP =  VAL (B$)
830  VTAB (6): IF S = 2 THEN 840
835  PRINT "NOTE AMOUNT BORROWED =$";
836 V = 6:H = 23: GOSUB 500:PV =  VAL (B$)
840  VTAB (7): IF S = 3 THEN 850
842  INPUT "# OF YEARS,MONTHS     = ";Y,M
844 Y = (Y * 12 + M) / 12:Y =  INT (Y * 100) / 100
850  VTAB (8): CALL  - 958: PRINT "PAYMENT PERIODS/YEAR
="
852  PRINT  TAB( 5)"1- YEARLY";: HTAB 25: PRINT "2-
SEMIANNUALLY"
854  PRINT  TAB( 5)"3- QUARTERLY";: HTAB 25: PRINT "4-
MONTHLY"
856  PRINT  TAB( 5)"5- DAILY"
860  VTAB (13): CALL  - 958: GOSUB 55
865  IF N > 0 AND N < 6 THEN 875
870  GOSUB 60: GOTO 860
875 C = PD(N)
880  RETURN
```

```
899  REM * MAIN DRIVER
900  HOME : PRINT : PRINT  TAB( 9)"*** LOAN SCHEDULES
***": PRINT
905  ONERR  GOTO 10000
910  PRINT "MENU:": PRINT "----"
912  FOR J = 1 TO 4: VTAB (5 + J): PRINT  TAB(
5)CH$(J): NEXT J
920  PRINT : PRINT  TAB( 5)"0- EXIT PROGRAM"
922  VTAB (13): CALL  - 958: GOSUB 55:S = N
924  IF S <  > 0 THEN 930
926  VTAB (22): CALL  - 958: PRINT "PROGRAM CALLED =
MAIN MENU"
927 P$ = "ID.000"
928  PRINT  CHR$ (4);"RUN ";P$;",D1"
930  IF S > 0 AND S < 5 THEN 934
932  GOSUB 60: GOTO 922
934  VTAB (5 + S): INVERSE : PRINT "*** ";CH$(S):
NORMAL
936  VTAB (15): CALL  - 958: INPUT "IS THAT
CORRECT?(Y/N) ";A$
938  IF  LEFT$ (A$,1) = "Y" THEN 942
940  IF  LEFT$ (A$,1) = "N" THEN 900
941  GOSUB 60: GOTO 936
942  ON S GOSUB 1000,2000,3000,4000
945  VTAB (20): CALL  - 958: INPUT "WANT NOTE SCHEDULE
DISPLAYED?(Y/N) ";A$
950  IF  LEFT$ (A$,1) = "Y" THEN 5000
955  IF  LEFT$ (A$,1) = "N" THEN 970
960  GOSUB 60: GOTO 945
970  VTAB (20): CALL  - 958: INPUT "WANT ANOTHER
CALCULATION?(Y/N) ";A$
975  IF  LEFT$ (A$,1) = "Y" THEN 900
980  IF  LEFT$ (A$,1) = "N" THEN 926
985  GOSUB 60: GOTO 970
1000  HOME : PRINT : PRINT  TAB( 6)"*** REGULAR NOTE
PAYMENT ***": PRINT
1010  GOSUB 800
1020  VTAB (4): CALL  - 958
1030  FOR J = 1 TO 4: ON J GOSUB 700,710,715,720: NEXT
J
1040 MP = (PV * ((I / C) / (1 - (1 + I / C) ^ ( - Y *
C))))
1050 MP =  INT (MP * 100 + 0.5) / 100
1060  PRINT : GOSUB 705
1070  HTAB 21: PRINT "---------"
1995  RETURN
```

```
2000  HOME : PRINT : PRINT  TAB( 5)"*** PRESENT VALUE
ON NOTE ***": PRINT
2010  GOSUB 800
2020  VTAB (4): CALL  - 958
2030  FOR J = 1 TO 4: ON J GOSUB 700,705,715,720: NEXT
J
2040 PV = MP * C / I:PV = PV * (1 - (1 / (1 + I / C) ^
(C * Y)))
2050 PV =  INT (PV * 100 + 0.5) / 100
2060  PRINT : GOSUB 710
2070  HTAB 21: PRINT "---------"
2995  RETURN
3000  HOME : PRINT : PRINT  TAB( 5)"*** # OF REGULAR
PAYMENTS ***": PRINT
3010  GOSUB 800
3020  VTAB (4): CALL  - 958
3030  FOR J = 1 TO 4: ON J GOSUB 705,710,715,720: NEXT
J
3040 Y1 =  LOG (1 / (1 - (PV * I) / (MP * C)))
3050 Y2 =  LOG (1 + I / C)
3060 Y = Y1 / Y2:Y =  INT (Y / 12 + 0.5)
3070  PRINT : GOSUB 700
3080  HTAB 21: PRINT "---------"
3995  RETURN
4000  HOME : PRINT : PRINT  TAB( 5)"*** ANNUAL INTEREST
RATE ***": PRINT
4010  GOSUB 800
4020  VTAB (4): CALL  - 958
4030  FOR J = 1 TO 4: ON J GOSUB 700,705,710,720: NEXT
J
4040 I = 1
4050  FOR J = 1 TO 100
4060 I1 = (MP * C / PV) * (1 - (1 + I / C) ^ ( - Y *
C))
4070  IF  ABS (I1 - I) <  = 0.00001 THEN 4100
4080 I = I1
4090  NEXT J
4100 I1 = I1 * 100:I =  INT (I1 * 100 + 0.5) / 10000
4120  PRINT : GOSUB 715
4130  HTAB 21: PRINT "---------"
4995  RETURN
```

```
5000  HOME : PRINT : PRINT  TAB( 9)"*** LOAN SCHEDULE
***"
5005 P = PV:TI = 0
5010  FOR J = 1 TO 5: ON J GOSUB 700,705,710,715,720:
NEXT J
5015  VTAB (9): PRINT "PMT# PRINCIPAL BALANCE  TOTAL
INTEREST"
5020  FOR J = 1 TO Y * C
5025  PRINT  TAB( 2)J;
5030 K2 = MP - (I * P / C): IF (K2 >  = P) THEN K2 = P
5035 K2 =  INT (K2 * 100 + 0.5) / 100: PRINT  TAB(
8)K2;
5040 K3 = MP - K2:K3 =  INT (K3 * 100 + 0.5) / 100
5045 P = P - K2: IF P <  = 0 THEN P = 0.0
5050 K4 =  INT (P * 100 + 0.5) / 100: PRINT  TAB(
17)K4;
5055 TI = TI + K3:TI =  INT (TI * 100 + 0.5) / 100:
PRINT  TAB( 28)TI
5060  IF K4 = 0 THEN 5070
5065  IF  INT (J / C) * C =  INT (J * C) / C THEN
GOSUB 100
5070  NEXT J
5075  GOSUB 100
5080  GOTO 970
10000 E =  PEEK (222): VTAB (22): CALL  - 958
10005  PRINT "ERROR = <";E;">";
10010  IF E = 255 THEN  PRINT " IS CRTL-C ": POKE
222,0: GOTO 11000
10020  IF E = 254 THEN  PRINT " IS DATA ERROR"
10500  GOSUB 65: CLEAR : RESTORE : GOTO 20
11000  VTAB (23): CALL  - 958: INPUT "DO YOU WISH TO
QUIT?(Y/N) = ";N$
11010  IF  LEFT$ (N$,1) = "Y" THEN 926
11020  IF  LEFT$ (N$,1) = "N" THEN 900
11030  GOSUB 60: GOTO 11000
```

圖12-3 貸款分期定額償還表程式

ID.020 = 複利終值表

```
0   POKE 1011,224
1   REM ***** ID.020 ***********
2   REM *
3   REM * WRITTEN BY:JL CAMPBELL
4   REM *       DATED:05/24/78
5   REM *
6   REM * COMPUTER:APPLE II
7   REM * LANGUAGE:APPLESOFT II
8   REM * SAVINGS PROGRAM
9   REM *******************
10   TEXT : CLEAR : POKE 216,0:A =  FRE (0)
12   DIM PD(5)
20   FOR I = 1 TO 5: READ PD(I): NEXT I
22   DATA  1,2,4,12,365
30   FOR I = 1 TO 4: READ CH$(I): NEXT I
32   DATA  1- FUTURE VALUE ON SAVINGS,2- PRESENT VALUE
ON SAVINGS,3- # OF PAYMENT PERIODS,4- ANNUAL INTEREST
RATE
50   GOTO 900
51   REM **************
52   REM * GEN'L SUBR *
53   REM **************
55   INPUT "WHICH ONE? ==> ";N: RETURN
60   VTAB (22): HTAB 10: INVERSE : PRINT "ILLEGAL
ENTRY": NORMAL : PRINT  CHR$ (7)
65   FOR L = 1 TO 1000: NEXT L: RETURN
100   VTAB (23): INPUT "TYPE RETURN KEY TO CONTINUE...
";A$
105   VTAB (10): CALL  - 958: RETURN
499   REM * NUMBER FILTER
500 IX = 1:B$ = "":SW = 0
505   VTAB (V): HTAB H + IX
510   PRINT "+";
515   VTAB (V): HTAB H + IX
520   GET X$
525   IF  ASC (X$) = 13 THEN 580
530   IF  ASC (X$) = 8 THEN 560
532   IF IX > 1 THEN 540
535   IF  ASC (X$) = 45 THEN 550
540   IF  ASC (X$) = 46 AND SW = 0 THEN SW = 1: GOTO 550
545   IF  ASC (X$) < 48 OR  ASC (X$) > 57 THEN 515
550 B$ = B$ + X$: PRINT X$
555 IX = IX + 1: GOTO 505
560   IF IX = 1 THEN 505
565   HTAB H + IX: PRINT " ";: HTAB H + IX - 1
567   IF  MID$ (B$, LEN (B$),1) = "." THEN SW = 0
```

```
570 B$ =  MID$ (B$,1, LEN (B$) - 1)
575 IX = IX - 1: GOTO 505
580  CALL  - 868
585  VTAB (V): HTAB H + 1: PRINT B$
590  RETURN
698  REM * SUMMARY DATA
700  PRINT "FUTURE VALUE OF ACCT=$";FV: RETURN
705  PRINT "ACC'T PRESENT VALUE =$";PV: RETURN
710  PRINT "# OF PAYMENT PERIODS= "; INT (Y * C):
RETURN
715  PRINT "THE ANNUAL INTEREST = ";I * 100;"%": RETURN

720  PRINT "COMPOUNDING PERIODS = ";C: RETURN
799  REM * INPUT DATA
800  VTAB (4): IF S = 1 THEN 820
805  PRINT "FUTURE VALUE OF ACCT =$";
806 V = 4:H = 23: GOSUB 500:FV =  VAL (B$)
820  VTAB (5): IF S = 2 THEN 830
825  PRINT "PRESENT VALUE OF ACCT=$";
826 V = 5:H = 23: GOSUB 500:PV =  VAL (B$)
830  VTAB (6): IF S = 3 THEN 840
835  INPUT "# OF YEARS,MONTHS     = ";Y,M
837 Y = (Y * 12 + M) / 12:Y =  INT (Y * 100) / 100
840  VTAB (7): IF S = 4 THEN 850
842  PRINT "ANNUAL INTEREST RATE = ";
843 V = 7:H = 23: GOSUB 500:I =  VAL (B$)
844 I = I / 100
850  VTAB (8): CALL  - 958: PRINT "# OF COMPOUNDING
PERIODS"
852  PRINT  TAB( 5)"1- YEARLY";: HTAB 25: PRINT "2-
SEMIANNUALLY"
854  PRINT  TAB( 5)"3- QUARTERLY";: HTAB 25: PRINT "4-
MONTHLY"
856  PRINT  TAB( 5)"5- DAILY"
860  VTAB (13): CALL  - 958: GOSUB 55
865  IF N > 0 AND N < 6 THEN 875
870  GOSUB 60: GOTO 860
875 C = PD(N)
895  RETURN
899  REM * MAIN DRIVER
900  HOME : PRINT : PRINT  TAB( 7)"*** SAVINGS
SCHEDULES ***": PRINT
905  ONERR  GOTO 10000
910  PRINT "MENU:": PRINT "----"
912  FOR J = 1 TO 4: PRINT  TAB( 5)CH$(J): NEXT J
```

```
920  PRINT : PRINT  TAB( 5)"0- EXIT PROGRAM"
922  VTAB (13): CALL  - 958: GOSUB 55:S = N
924  IF S <  > 0 THEN 930
926  VTAB (22): CALL  - 958: PRINT "PROGRAM CALLED =
MAIN MENU"
927 P$ = "ID.000"
928  PRINT  CHR$ (4);"RUN ";P$;",D1"
930  IF S > 0 AND S < 5 THEN 934
932  GOSUB 60: GOTO 922
934  VTAB (5 + S): INVERSE : PRINT "*** ";CH$(S):
NORMAL
936  VTAB (15): CALL  - 958: INPUT "IS THAT
CORRECT?(Y/N) ";A$
938  IF  LEFT$ (A$,1) = "Y" THEN 942
940  IF  LEFT$ (A$,1) = "N" THEN 900
941  GOSUB 60: GOTO 936
942  ON S GOSUB 1000,2000,3000,4000
970  VTAB (20): CALL  - 958: INPUT "WANT ANOTHER
CALCULATION?(Y/N) ";A$
975  IF  LEFT$ (A$,1) = "Y" THEN 900
980  IF  LEFT$ (A$,1) = "N" THEN 926
985  GOSUB 60: GOTO 970
1000  HOME : PRINT : PRINT  TAB( 10)"*** FUTURE VALUE
***": PRINT
1010  GOSUB 800
1020  VTAB (4): CALL  - 958
1030  FOR J = 1 TO 4: ON J GOSUB 705,710,715,720: NEXT
J
1040 FV = PV * ((1 + I / C) ^ (Y * C))
1050 FV =  INT (FV * 100 + 0.5) / 100
1060  PRINT : GOSUB 700
1070  HTAB 22: PRINT "---------"
1995  RETURN
2000  HOME : PRINT : PRINT  TAB( 9)"*** PRESENT VALUE
***": PRINT
2010  GOSUB 800
2020  VTAB (4): CALL  - 958
2030  FOR J = 1 TO 4: ON J GOSUB 700,710,715,720: NEXT
J
2040 PV = FV * ((1 + I / C) ^ ( - Y * C))
2050 PV =  INT (PV * 100 + 0.5) / 100
2060  PRINT : GOSUB 705
2070  HTAB 22: PRINT "---------"
2995  RETURN
```

```
3000  HOME : PRINT : PRINT  TAB( 6)"*** # OF REGULAR
PERIODS ***": PRINT
3010  GOSUB 800
3020  VTAB (4): CALL  - 958
3030  FOR J = 1 TO 4: ON J GOSUB 700,705,715,720: NEXT
J
3040 Y1 =  LOG (FV / PV)
3050 Y2 =  LOG (1 + I / C)
3060 Y = Y1 / (Y2 * C):Y =  INT (Y * 100 + 0.5) / 100
3070  PRINT : GOSUB 710
3080  HTAB 22: PRINT "----------"
3995  RETURN
4000  HOME : PRINT : PRINT  TAB( 5)"*** ANNUAL INTEREST
RATE ***": PRINT
4010  GOSUB 800
4020  VTAB (4): CALL  - 958
4030  FOR J = 1 TO 4: ON J GOSUB 700,705,710,720: NEXT
J
4040 I = C * ((FV / PV) ^ (1 / (Y * C)) - 1)
4060 I =  INT (I * 100 + 0.5) / 100
4120  PRINT : GOSUB 715
4130  HTAB 22: PRINT "----------"
4995  RETURN
10000 E =  PEEK (222): VTAB (22): CALL  - 958
10005  PRINT "ERROR = <";E;">";
10010  IF E = 255 THEN  PRINT " IS CRTL-C ": POKE
222,0: GOTO 11000
10020  IF E = 254 THEN  PRINT " IS DATE ERROR"
10500  GOSUB 65: CLEAR : RESTORE : GOTO 20
11000  VTAB (23): CALL  - 958: INPUT "DO YOU WISH TO
QUIT?(Y/N) = ";N$
11010  IF  LEFT$ (N$,1) = "Y" THEN 926
11020  IF  LEFT$ (N$,1) = "N" THEN 900
11030  GOSUB 60: GOTO 11000
```

圖12-4 複利終值表程式

ID.030 = 年金終値表

```
0   POKE 1011,224
1   REM ***** ID.030 ***********
2   REM *
3   REM * WRITTEN BY:JL CAMPBELL
4   REM *       DATED:05/24/78
5   REM *
6   REM * COMPUTER:APPLE II
7   REM * LANGUAGE:APPLESOFT II
8   REM * ANNUITY SCHEDULES
9   REM ******************
10   TEXT : CLEAR : POKE 216,0:A =  FRE (0)
12   DIM PD(5),CH$(4)
20   FOR I = 1 TO 5: READ PD(I): NEXT I
22   DATA  1,2,4,12,365
30   FOR I = 1 TO 4: READ CH$(I): NEXT I
32   DATA  1- FUTURE VALUE OF ANNUITY,2- PAYMENT PER
PERIOD,3- # OF PAYMENT PERIODS
34   DATA  4- ANNUAL INTEREST RATE
50   GOTO 900
51   REM **************
52   REM * GEN'L SUBR *
53   REM **************
55   INPUT "WHICH ONE? ==> ";N: RETURN
60   VTAB (22): HTAB 10: INVERSE : PRINT "ILLEGAL
      ENTRY": NORMAL : PRINT  CHR$ (7)
65   FOR L = 1 TO 1000: NEXT L: RETURN
70   HTAB 22: PRINT "----------": RETURN
100   VTAB (23): INPUT "TYPE RETURN KEY TO CONTINUE..
      . ";A$
105   VTAB (10): CALL  - 958: RETURN
499   REM * NUMBER FILTER
500 IX = 1:B$ = "":SW = 0
505   VTAB (V): HTAB H + IX
510   PRINT "+";
515   VTAB (V): HTAB H + IX
520   GET X$
525   IF  ASC (X$) = 13 THEN 580
530   IF  ASC (X$) = 8 THEN 560
532   IF IX > 1 THEN 540
535   IF  ASC (X$) = 45 THEN 550
540   IF  ASC (X$) = 46 AND SW = 0 THEN SW = 1: GOTO 550
545   IF  ASC (X$) < 48 OR  ASC (X$) > 57 THEN 515
550 B$ = B$ + X$: PRINT X$
555 IX = IX + 1: GOTO 505
```

```
560  IF IX = 1 THEN 505
565  HTAB H + IX: PRINT " ";: HTAB H + IX - 1
567  IF  MID$ (B$, LEN (B$),1) = "." THEN SW = 0
570 B$ =  MID$ (B$,1, LEN (B$) - 1)
575 IX = IX - 1: GOTO 505
580  CALL  - 868
585  VTAB (V): HTAB H + 1: PRINT B$
590  RETURN
698  REM * SUMMARY DATA
700  PRINT "FUTURE VALUE OF ACCT=$";FV: RETURN
705  PRINT "PAYMENT EACH PERIOD =$";PV: RETURN
710  PRINT "# OF PAYMENT PERIODS= "; INT (Y * C):
RETURN
715  PRINT "THE ANNUAL INTEREST = ";I * 100;"%": RETURN

720  PRINT "PAYMENTS EACH YEAR  = ";C: RETURN
799  REM * INPUT DATA
800  VTAB (4): IF S = 1 THEN 820
805  PRINT "FUTURE VALUE OF ACCT =$";
806 V = 4:H = 23: GOSUB 500:FV =  VAL (B$)
820  VTAB (5): IF S = 2 THEN 830
825  PRINT "PAYMENT EACH PERIOD  =$";
826 V = 5:H = 23: GOSUB 500:PV =  VAL (B$)
830  VTAB (6): IF S = 3 THEN 840
835  INPUT "# OF YEARS,MONTHS    = ";Y,M
837 Y = (Y * 12 + M) / 12:Y =  INT (Y * 100) / 100
840  VTAB (7): IF S = 4 THEN 850
842  PRINT "ANNUAL INTEREST RATE = ";
843 V = 7:H = 23: GOSUB 500:I =  VAL (B$)
844 I = I / 100
850  VTAB (8): CALL  - 958: PRINT "# OF PAYMENTS/YEAR
="
852  PRINT  TAB( 5)"1- YEARLY";: HTAB 25: PRINT "2-
SEMIANNUALLY"
854  PRINT  TAB( 5)"3- QUARTERLY";: HTAB 25: PRINT "4-
MONTHLY"
856  PRINT  TAB( 5)"5- DAILY"
860  VTAB (14): CALL  - 958: GOSUB 55
865  IF N > 0 AND N < 6 THEN 875
870  GOSUB 60: GOTO 860
875 C = PD(N)
895  RETURN
```

```
899  REM * MAIN DRIVER
900  HOME : PRINT : PRINT  TAB( 7)"*** ANNUITY
SCHEDULES ***": PRINT
905  ONERR  GOTO 10000
910  PRINT "MENU:": PRINT "----"
912  FOR J = 1 TO 4: PRINT  TAB( 5)CH$(J): NEXT J
920  PRINT : PRINT  TAB( 5)"0- EXIT PROGRAM"
922  VTAB (13): CALL  - 958: GOSUB 55:S = N
924  IF S <  > 0 THEN 930
926  VTAB (22): CALL  - 958: PRINT "PROGRAM CALLED =
MAIN MENU"
927 P$ = "ID.000"
928  PRINT  CHR$ (4);"RUN ";P$;",D1"
930  IF S > 0 AND S < 5 THEN 934
932  GOSUB 60: GOTO 922
934  VTAB (5 + S): INVERSE : PRINT "*** ";CH$(S):
NORMAL
936  VTAB (15): CALL  - 958: INPUT "IS THAT
CORRECT?(Y/N) ";A$
938  IF  LEFT$ (A$,1) = "Y" THEN 942
940  IF  LEFT$ (A$,1) = "N" THEN 900
941  GOSUB 60: GOTO 936
942  ON S GOSUB 1000,2000,3000,4000
944  IF S = 3 OR S = 4 THEN 970
950  VTAB (22): CALL  - 958: INPUT "WANT PRINTED
      REPORT?(Y/N) ";A$
952  IF  LEFT$ (A$,1) = "Y" THEN 5000
954  IF  LEFT$ (A$,1) = "N" THEN 970
956  GOSUB 60: GOTO 950
970  VTAB (22): CALL  - 958: INPUT "WANT ANOTHER
      CALCULATION?(Y/N) ";A$
975  IF  LEFT$ (A$,1) = "Y" THEN 900
980  IF  LEFT$ (A$,1) = "N" THEN 926
985  GOSUB 60: GOTO 970
1000  HOME : PRINT : PRINT  TAB( 10)"*** FUTURE VALUE
       ***": PRINT
1010  GOSUB 800
1020  VTAB (4): CALL  - 958
1030  FOR J = 1 TO 4: ON J GOSUB 705,710,715,720:
       NEXT J
1040 FV = PV * (1 + I / C) * ((1 + I / C) ^ (Y * C)
      - 1) / (I / C)
1050 FV =  INT (FV * 100 + 0.5) / 100
1060  PRINT : GOSUB 700
1070  GOSUB 70
```

```
1100  VTAB (12): CALL  - 958: INPUT "FRONT LOADED?(Y/N)
       = ";A$
1120  IF  LEFT$ (A$,1) = "Y" THEN 1200
1130  IF  LEFT$ (A$,1) = "N" THEN 1500
1140  GOSUB 60: GOTO 1100
1200  VTAB (13): CALL  - 958: PRINT "PERCENT LOADING
       (%) = ";
1210 V = 13:H = 22: GOSUB 500:PL =  VAL (B$)
1260 PL = PL / 100
1270 P1 = PV + PV * PL:P1 =  INT (P1 * 100 + 0.5) / 100
1280  VTAB (15): PRINT "NEW PAYMENT/PERIOD  =$";P1
1290  GOSUB 70
1300  PRINT  TAB( 5)"***** OR *****"
1310 P2 = PV - PL * PV
1320 F2 = P2 * (1 + I / C) * ((1 + I / C) ^ (Y * C)
       - 1) / (I / C)
1330 F2 =  INT (F2 * 100 + 0.5) / 100
1335  PRINT
1340  PRINT "NEW FUTURE VALUE     =$";F2
1350  GOSUB 70
1500  RETURN
2000  HOME : PRINT : PRINT  TAB( 9)"*** PAYMENT/PERIOD
       ***": PRINT
2010  GOSUB 800
2020  VTAB (4): CALL  - 958
2030  FOR J = 1 TO 4: ON J GOSUB 700,710,715,720:
       NEXT J
2040 PV = FV / (1 + I / C)
2045 PV = PV * ((I / C) / ((1 + I / C) ^ (Y * C) - 1))
2050 PV =  INT (PV * 100 + 0.5) / 100
2060  PRINT : GOSUB 705
2070  GOSUB 70
2100  VTAB (12): CALL  - 958
2110  INPUT "FRONT LOADED?(Y/N)  = ";FL$
2120  IF  LEFT$ (FL$,1) = "Y" THEN 2200
2130  IF  LEFT$ (FL$,1) = "N" THEN 2500

2140  GOSUB 60: GOTO 2100
2200  VTAB (13): CALL  - 958: PRINT "PERCENT LOADING
       (%) = ";
2210 V = 13:H = 22: GOSUB 500:PL =  VAL (B$)
2260 PL = PL / 100
2270 P1 = PV + PV * PL:P1 =  INT (P1 * 100 + 0.5) / 100
2280  VTAB (15): PRINT "NEW PAYMENT/PERIOD  =$";P1
2290  GOSUB 70
2300  PRINT  TAB( 5)"***** OR *****"
2310 P2 = PV - PL * PV
2320 F2 = P2 * (1 + I / C) * ((1 + I / C) ^ (Y * C)
       - 1) / (I / C)
2330 F2 =  INT (F2 * 100 + 0.5) / 100
2335  PRINT
```

```
2340  PRINT "NEW FUTURE VALUE    =$";F2
2350  GOSUB 70
2500  RETURN
3000  HOME : PRINT : PRINT  TAB( 6)"*** # OF REGULAR
        PERIODS ***": PRINT
3010  GOSUB 800
3020  VTAB (4): CALL  - 958
3030  FOR J = 1 TO 4: ON J GOSUB 700,705,715,720:
        NEXT J
3040 Y1 =  LOG (1 + (FV * (I / C)) / ((PV * (1 +
        I / C))))
3050 Y2 =  LOG (1 + (I / C))
3060 Y = Y1 / Y2:Y = Y / C
3065 Y =  INT (Y * 100 + 0.5) / 100
3070  PRINT : GOSUB 710
3080  GOSUB 70
3995  RETURN
4000  HOME : PRINT : PRINT  TAB( 5)"*** ANNUAL INTEREST
        RATE ***": PRINT
4010  GOSUB 800
4020  VTAB (4): CALL  - 958
4030  FOR J = 1 TO 4: ON J GOSUB 700,705,710,720:
        NEXT J
4040 H = 1:L = 0
4045 I = (H + L) / 2
4050 X = PV * (1 + I / C) * ((1 + I / C) ^ (Y * C)
        - 1) / (I / C)
4055  IF  ABS (X - FV) <  = 10 THEN 4100
4060  IF X > FV THEN 4070
4065 L = I: GOTO 4045
4070 H = I: GOTO 4045
4100 I = I * 100:I =  INT (I * 100 + 0.5) / 10000
4120  PRINT : GOSUB 715
4130  GOSUB 70
4995  RETURN
```

```
5000  HOME : PRINT : PRINT  TAB( 9)"*** PRINTED REPORT
        ***": PRINT
5010  PRINT "DO THE FOLLOWING:"
5020  PRINT  TAB( 5)"1- LOAD PRINTER PAPER"
5030  PRINT  TAB( 5)"2- TURN ON PRINTER"
5040  PRINT  TAB( 5)"3- SET TOP-OF-FORM"
5050  PRINT  TAB( 5)"4- PRINTER ON-LINE"
5060  PRINT : INPUT "TYPE RETURN KEY TO
       CONTINUE....";A$
5070  VTAB 12: CALL  - 958: INPUT "PRINTER
       SLOT?(1-7) ";SL
5072  IF SL > 0 AND SL < 8 THEN 5080
5074  GOSUB 60: GOTO 5070
5080  VTAB 14: CALL  - 958: PRINT "PRINTER TYPE:"
5082  PRINT  TAB( 5)"1- SERIAL"
5084  PRINT  TAB( 5)"2- PARALLEL"
5086  PRINT : GOSUB 55:SP = N
5088  IF SP = 1 OR SP = 2 THEN 5100
5090  GOSUB 60: GOTO 5080
5100  PR# SL
5110  IF SP = 2 THEN  PRINT  CHR$ (9);"80N"
5120  PRINT  TAB( 33)"ANNUITY REPORT"
5130  PRINT " "
5140  PRINT  TAB( 10)"FUTURE VALUE OF ACC'T=$";FV
5150  PRINT  TAB( 10)"PAYMENT EACH PERIOD  =$";PV
5160  PRINT  TAB( 10)"# OF PAYMENT PERIODS = ";
       INT (Y * C)
5170  PRINT  TAB( 10)"THE ANNUAL INTEREST  = ";
        I * 100;" %"
5180  PRINT  TAB( 10)"PAYMENTS EACH YEAR   = ";C
5190  PRINT " "
5200  IF PL = 0 THEN 5470
5210  PRINT  TAB( 10)"PERCENT LOADING (%)  = ";
        INT (PL * 10000 + 0.5) / 100
5220  PRINT " "
5240  PRINT  TAB( 10)"NEW PAYMENT/PERIOD   =$";P1
5245  PRINT  TAB( 33)"---------"
5250  PRINT " "
5260  PRINT  TAB( 15)"*** OR ***"
5270  PRINT " "
5280  PRINT  TAB( 10)"NEW FUTURE VALUE     =$";F2
5285  PRINT  TAB( 33)"---------"
```

```
5470  PRINT  CHR$ (12)
5480  IF SP = 2 THEN  PRINT  CHR$ (9);"40N"
5490  PR# 0
5500  GOTO 970
10000 E =  PEEK (222): VTAB (22): CALL  - 958
10005  PRINT "ERROR = <";E;">";
10010  IF E = 255 THEN  PRINT " IS CRTL-C ":
         POKE 222,0: GOTO 11000
10020  IF E = 254 THEN  PRINT " IS DATA ERROR"
10500  GOSUB 65: CLEAR : RESTORE : GOTO 20
11000  VTAB (23): CALL  - 958: INPUT "DO YOU WISH TO
         QUIT?(Y/N) = ";N$
11010  IF  LEFT$ (N$,1) = "Y" THEN 926
11020  IF  LEFT$ (N$,1) = "N" THEN 900
11030  GOSUB 60: GOTO 11000
```

圖12-5　年金終値表程式

ID.040 = 計算折舊程式

```
0   POKE 1011,224
1   REM ***** ID.040 ***********
2   REM *
3   REM * WRITTEN BY:JL CAMPBELL
4   REM *        DATED:07/17/80
5   REM *
6   REM * COMPUTER:APPLE II
7   REM * LANGUAGE:APPLESOFT II
8   REM * DEPRECIATION PROGRAM
9   REM *******************
10   TEXT : CLEAR : POKE 216,0:A =  FRE (0)
20   DIM D(30,5),T(4),D$(30,4),T$(4),Z(8)
50   GOTO 900
51   REM **************
52   REM * GEN'L SUBR *
53   REM **************
60   VTAB (22): HTAB 10: INVERSE : PRINT "ILLEGAL
      ENTRY": NORMAL : PRINT  CHR$ (7)
65   FOR L = 1 TO 1000: NEXT L: RETURN
99   REM * MAKE STRING
100   FOR K = 1 TO 8:Z(K) = 0: NEXT K
105 X1 =  INT (X * 100 + 0.5)
110   FOR K = 8 TO 1 STEP  - 1
115 X2 =  INT (X1 / 10):X3 =  INT (X2 * 10)
120 Z(K) =  INT (X1 - X3)
125 X1 = X2
130   NEXT K
135 X$ = "":SW = 0
140   FOR K = 1 TO 6
145   IF SW = 0 AND Z(K) = 0 THEN X$ = X$ + " ": GOTO
160
150 X$ = X$ +  STR$ (Z(K))
155 SW = 1
160   NEXT K
165 X$ = X$ + "." +  STR$ (Z(7)) +  STR$ (Z(8))
170   RETURN
499   REM * NUMBER FILTER
500 IX = 1:B$ = "":SW = 0
505   VTAB V: HTAB H + IX
510   PRINT "+";
515   VTAB V: HTAB H + IX
520   GET X$
525   IF  ASC (X$) = 13 THEN 580
530   IF  ASC (X$) = 8 THEN 560
532   IF IX > 1 THEN 540
535   IF  ASC (X$) = 45 THEN 550
```

```
540  IF  ASC (X$) = 46 AND SW = 0 THEN SW = 1: GOTO 550
545  IF  ASC (X$) < 48 OR  ASC (X$) > 57 THEN 515
550 B$ = B$ + X$: PRINT X$
555 IX = IX + 1: GOTO 505
560  IF IX = 1 THEN 505
565  HTAB H + IX: PRINT " ";: HTAB H + IX - 1
567  IF  MID$ (B$, LEN (B$),1) = "." THEN SW = 0
570 B$ =  MID$ (B$,1, LEN (B$) - 1)
575 IX = IX - 1: GOTO 505
580  CALL  - 868
585  VTAB V: HTAB H + 1: PRINT B$
590  RETURN
899  REM * MAIN DRIVER
900  HOME : PRINT : PRINT  TAB( 10)"*** DEPRECIATION
      ***": PRINT
910  PRINT "PURCHASE PRICE =$";:V = 4:H = 17
911  GOSUB 500:PP =  VAL (B$)
912  PRINT "SALVAGE VALUE  =$";:V = 5
913  GOSUB 500:SV =  VAL (B$)
914  PRINT "USEFUL YEARS   = ";:V = 6
915  GOSUB 500:YR =  VAL (B$)
916  PRINT "USEFUL HOURS   = ";:V = 7
917  GOSUB 500:HR =  VAL (B$)
920  VTAB 12: HTAB 10: INVERSE : PRINT ".....
      PROCESSING.....": NORMAL
1000  REM * STRAIGHT LINE
1020 P1 = PP:S1 = SV:T(1) = 0
1030 C1 = SV / PP: IF C1 <  = 0.1 THEN S1 = 0
1040 SL = (P1 - S1) / YR:SL =  INT (SL * 100 + 0.5) /
100
1050  FOR I = 1 TO YR
1060 D(I,1) = SL:T(1) = T(1) + D(I,1)
1070  NEXT I
1080 T(1) =  INT (T(1) * 100 + 0.5) / 100
2000  REM * DOUBLE DECLINING
2020 P2 = PP:S2 = SV:C2 = 1 / YR:T(2) = 0
2030  FOR I = 1 TO YR
2040 D(I,2) = 2 * C2 * P2:P2 = P2 - D(I,2):
      T(2) = T(2) + D(I,2)
2050 D(I,2) =  INT (D(I,2) * 100 + 0.5) / 100
2060  NEXT I
2070 T(2) =  INT (T(2) * 100 + 0.5) / 100
3000  REM * SUM-OF-YEARS-DIGITS
3020 P3 = PP:S3 = SV:T(3) = 0:SD = 0
3030  FOR I = 1 TO YR:SD = SD + I: NEXT I
3040  FOR I = 1 TO YR
```

```
3050 D(YR + 1 - I,3) = I / SD * P3:T(3)
      = T(3) + D(YR + 1 - I,3)
3060 D(YR + 1 - I,3) =  INT (D(YR + 1 - I,3) *
      100 + 0.5) / 100
3080  NEXT I
3090 T(3) =  INT (T(3) * 100 + 0.5) / 100
4000  REM * UNITS OF PRODUCTION
4010 T(4) = 0:C4 = SV / PP:S4 = SV
4015  IF C4 <  = 0.1 THEN S4 = 0
4020 D4 = (PP - S4) / HR
4030  FOR I = 1 TO YR
4040 D(I,4) = D4 * HR / YR
4050 T(4) = T(4) + D(I,4)
4060  NEXT I
4900  FOR I = 1 TO YR: FOR J = 1 TO 4
4910 X = D(I,J): GOSUB 100:D$(I,J) = X$
4920  NEXT J,I
4930  FOR I = 1 TO 4
4940 X = T(I): GOSUB 100:T$(I) = X$
4950  NEXT I
5000  VTAB 9: CALL  - 958
5010  PRINT "YR     SL      DDB      SOD      UOP"
5020  FOR I = 1 TO 38: PRINT "-";: NEXT I: PRINT
5030  FOR I = 1 TO YR
5035  PRINT I;
5040  FOR J = 1 TO 4: HTAB 4 + (J - 1) * 9:
        PRINT D$(I,J);: NEXT J: PRINT
5045  IF I = YR THEN 5110
5050  IF  INT (I / 5) * 5 <  >  INT (I * 5) / 5 THEN
5090
5060  VTAB (20): INPUT "TYPE RETURN TO CONTINUE...";A$
5070  VTAB (11): CALL  - 958
5090  NEXT I
5110  FOR I = 1 TO 4: HTAB 4 + (I - 1) * 9:
       PRINT "---------";: NEXT I: PRINT
5120  FOR I = 1 TO 4: HTAB 4 + (I - 1) * 9:
       PRINT T$(I);: NEXT I: PRINT
5200  VTAB (20): CALL  - 958: INPUT "WANT PRINTED
       REPORT?(Y/N) ";A$
5210  IF  LEFT$ (A$,1) = "Y" THEN 5240
5220  IF  LEFT$ (A$,1) = "N" THEN 9700
5230  GOSUB 60: GOTO 5200
5240  VTAB (4): CALL  - 958: PRINT "DO THE FOLLOWING;"
5250  PRINT  TAB( 5)"1- LOAD PRINTER PAPER"
5255  PRINT  TAB( 5)"2- TURN ON PRINTER"
5260  PRINT  TAB( 5)"3- SET TOP-OF-FORM"
5270  PRINT  TAB( 5)"4- PRINTER ON-LINE"
5280  PRINT : INPUT "TYPE RETURN KEY
        TO CONTINUE....";A$
```

```
5290  VTAB (12): CALL  - 958: INPUT "PRINTER
       SLOT(1-7) ==> ";SL
5295  IF SL > 0 AND SL < 8 THEN 5305
5300  GOSUB 60: GOTO 5290
5305  VTAB (14): CALL  - 958: PRINT "PRINTER TYPE:"
5310  PRINT  TAB( 5)"1- SERIAL"
5315  PRINT  TAB( 5)"2- PARALLEL"
5320  PRINT : INPUT "WHICH ONE ==> ";SP
5325  IF SP = 1 OR SP = 2 THEN 5360
5330  GOSUB 60: GOTO 5305
5360  PR# SL
5370  IF SP = 2 THEN  PRINT  CHR$ (9);"132N"
5380  PRINT  TAB( 30)"DEPRECIATION REPORT": PRINT " "
5390 X = PP: GOSUB 100
5400  PRINT  TAB( 10)"PURCHASE PRICE = $";X$
5410 X = SV: GOSUB 100
5420  PRINT  TAB( 10)"SALVAGE VALUE  = $";X$
5430  PRINT  TAB( 10)"USEFUL YEARS   =  ";YR
5440  PRINT  TAB( 10)"USEFUL HOURS   =  ";HR
5450  PRINT " "
5460  FOR I = 1 TO 78: PRINT "=";: NEXT I:
        PRINT " ": PRINT " "
5470  PRINT "YEAR","     SL","   DDB","  SOD"," UOP"
5480  FOR I = 1 TO YR: PRINT I;
5500  FOR J = 1 TO 4: POKE 36,(17 + 15 * (J - 1))
5510  PRINT D$(I,J);
5520  NEXT J: PRINT " "
5530  NEXT I: PRINT " "
5540  FOR J = 1 TO 4: POKE 36,(16 + 15 * (J - 1)):
       PRINT "----------";: NEXT J: PRINT " "
5550  FOR I = 1 TO 4: POKE 36,(17 + 15 * (I - 1))
5560  PRINT T$(I),
5570  NEXT I: PRINT " "
5580  PRINT  CHR$ (12)
6000  IF SP = 2 THEN  PRINT  CHR$ (9);"40N"
6010  PR# 0
9700  VTAB (20): CALL  - 958: INPUT "WANT ANOTHER
       CALCULATION?(Y/N) ";A$
9710  IF  LEFT$ (A$,1) = "Y" THEN 900
9720  IF  LEFT$ (A$,1) = "N" THEN 9740
9730  GOSUB 60: GOTO 9700
9740  VTAB (22): CALL  - 958: PRINT "PROGRAM CALLED
       = MAIN MENU"
9750 P$ = "ID.000"
9760  PRINT  CHR$ (4);"RUN ";P$;",D1"
```

圖12-6 計算折舊程式

ID.050 = 攤銷表程式

```
0   POKE 1011,224
1   REM   ***** ID.050 **********
2   REM   *
3   REM   * WRITTEN BY:JL CAMPBELL
4   REM   *      DATED:08/30/79
5   REM   *
6   REM   * COMPILER:APPLESOFT II
7   REM   * COMPUTER:APPLE II
8   REM   * AMORTIZATION PROG
9   REM   * ADAPTED FROM: "SOME
10   REM * COMMON BASIC PROGRAMS"
11   REM *
12   REM ***********************
14   TEXT : CLEAR : POKE 216,0:A =  FRE (0)
20   DIM S(9),C(12),QQ(5)
30   FOR J = 1 TO 12: READ C(J): NEXT J
32   DATA  31,28,31,30,31,30,31,31,30,31,30,31
40 C$ =  CHR$ (9):BL$ = " ":F$ = " "
42 D1$ = " ":D2$ = " ":D3$ = " ":D4$ = " "
44   FOR J = 1 TO 4: READ QQ(J): NEXT J
46   DATA  12,6,3,1
50   GOTO 1000
51   REM **************
52   REM * GEN'L SUBR *
53   REM **************
55   REM * DATE SUBR
60   IF SM% <  =  INT (12) THEN 75
65 SM% = SM% - (12 + ST%): IF N = 12 THEN SM% = 0
70 SY% = SY% + 1
75 SM% = SM% + ST%
80   RETURN
90   VTAB (22): HTAB 10: INVERSE : PRINT "ILLEGAL
ENTRY"; CHR$ (7): NORMAL
95   FOR L = 1 TO 1000: NEXT L: RETURN
200   REM * DECOMPOSE NUMBER
205 NR =  INT (NR * 100 + 0.5)
210   FOR J = 9 TO 1 STEP  - 1
215 Q =  INT (NR / 10):Z =  INT (Q * 10)
220 S(J) =  INT (NR - Z)
225 NR = Q
230   NEXT J
235   RETURN
300   REM * MAKE STRING
305 F$ = "":SW = 0
310 F$ = D$
315   FOR J = 1 TO 4
```

```
320  IF SW = 0 AND S(J) = 0 THEN F$ = F$ + BL$: GOTO
335
325 F$ = F$ +  STR$ (S(J))
330 SW = 1
335  NEXT J
340  IF SW = 0 THEN F$ = F$ + BL$: GOTO 350
345 F$ = F$ + ","
350  FOR J = 5 TO 7
355  IF SW = 0 AND S(J) = 0 THEN F$ = F$ + BL$: GOTO
370
360 F$ = F$ +  STR$ (S(J))
365 SW = 1
370  NEXT J
375 F$ = F$ + "." +  STR$ (S(8)) +  STR$ (S(9))
380  IF D$ = BL$ THEN 398
385  FOR J = 2 TO 8
390  IF  MID$ (F$,J,1) <  > BL$ THEN 396
392 JX = J
394  NEXT J
396 E$ =  MID$ (F$,2,JX - 1) + D$ +  MID$ (F$,JX + 1,
LEN (F$)):F$ = E$
398  RETURN
399  REM * STRING FILTER
400 IX = 1:B$ = ""
405  VTAB (V): HTAB H + IX
410  PRINT "+";
415  VTAB (V): HTAB H + IX
420  GET X$
425  IF  ASC (X$) = 13 THEN 470
430  IF  ASC (X$) = 8 THEN 450
432  IF  ASC (X$) = 32 THEN 440
434  IF  ASC (X$) = 46 THEN 440
435  IF  ASC (X$) < 65 OR  ASC (X$) > 90 THEN 415
440 B$ = B$ + X$: PRINT X$
445 IX = IX + 1: GOTO 405
450  IF IX = 1 THEN 405
455  HTAB H + IX: PRINT " ";: HTAB H + IX - 1
460 B$ =  MID$ (B$,1, LEN (B$) - 1)
465 IX = IX - 1: GOTO 405
470  CALL  - 868
475  VTAB (V): HTAB H + 1: PRINT B$
480  RETURN
499  REM * NUMBER FILTER
500 IX = 1:B$ = "":SW = 0
505  VTAB (V): HTAB H + IX
510  PRINT "+";
515  VTAB (V): HTAB H + IX
520  GET X$
525  IF  ASC (X$) = 13 THEN 580
530  IF  ASC (X$) = 8 THEN 560
```

```
532  IF IX > 1 THEN 540
535  IF  ASC (X$) = 45 THEN 550
540  IF  ASC (X$) = 46 AND SW = 0 THEN SW = 1: GOTO 550
545  IF  ASC (X$) < 48 OR  ASC (X$) > 57 THEN 515
550 B$ = B$ + X$: PRINT X$
555 IX = IX + 1: GOTO 505
560  IF IX = 1 THEN 505
565  HTAB H + IX: PRINT " ";: HTAB H + IX - 1
567  IF  MID$ (B$, LEN (B$),1) = "." THEN SW = 0
570 B$ =  MID$ (B$,1, LEN (B$) - 1)
575 IX = IX - 1: GOTO 505
580  CALL  - 868
585  VTAB (V): HTAB H + 1: PRINT B$
590  RETURN
700  REM * HEADER SUBR
705  PRINT  TAB( 66)"PAGE ";PG%: PRINT " "
710  PRINT  TAB( 26)"MORTGAGE AMORTIZATION TABLE":
PRINT  TAB( 65)"DATE: ";DY$
715  PRINT  TAB( 3)"BUYER: ";NM$;: POKE 36,43: PRINT
"SELLER: ";SN$
720  PRINT  TAB( 3)"PRINCIPAL $";P;" AT ";I * 100;"%
FOR "; INT (Y);" YEARS "; INT (M);" MONTHS."
725  PRINT  TAB( 3)"REGULAR PAYMENT = $";R
730  PRINT " "
735  PRINT "DATE","    INTEREST","   PRINCIPAL","
BALANCE","TOTAL INTEREST"
740  FOR L = 1 TO 78: PRINT "-";: NEXT L: PRINT " "
745 K = 6:D$ = "$"
750  RETURN
800  REM * YEAR END SUBR
805 D$ = "$"
810  PRINT " "," -----------"," -----------"
815 NR = I3: GOSUB 200: GOSUB 300:D5$ = F$
820 NR = A2: GOSUB 200: GOSUB 300:D6$ = F$
825  PRINT "TOTAL ";SY%,D5$,D6$
830  PRINT " "
835  RETURN
900  REM * LAST PYMT SUBR
905 D$ = "$"
910 NR =  INT (R * 100 + 0.5) / 100
915  GOSUB 200: GOSUB 300:D7$ = F$
920  PRINT "LAST PAYMENT =",D7$
925  RETURN
1000  HOME : PRINT : PRINT  TAB( 5)"MORTGAGE
AMORTIZATION TABLE": PRINT
1002  PRINT "ENTER TODAYS DATE(MMDDYY) ";
1003 V = 4:H = 26: GOSUB 500:D$ = B$
1004  IF B$ = "" AND IX <  = 2 THEN 1018
1006 L =  LEN (D$): IF L = 6 THEN 1010
```

```
1003 C 2) = 28: GOSUB 90: GOTO 1000
1010 Y =  VAL ( MID$ (D$,5,2)):D =  VAL ( MID$
(D$,3,2)):M =  VAL ( LEFT$ (D$,2))
1011  IF  INT (Y / 4) = Y / 4 THEN C(2) = 29
1012  IF M > 0 AND M < 13 THEN 1014
1013  GOTO 1008
1014  IF D > 0 AND D <  = C(M) THEN 1016
1015  GOTO 1008
1016 DY$ =  LEFT$ (D$,2) + "/" +  MID$ (D$,3,2) + "/" +
MID$ (D$,5.2)
1018 D$ = "$"
1020  VTAB (8): PRINT "REGULAR PAYMENT =$";
1022 V = 8:H = 18: GOSUB 500:R =  VAL (B$)
1030  VTAB (9): INPUT "TERM IN YRS,MOS.= ";Y,M
1032  VTAB (10): PRINT "PRINCIPAL AMOUNT=$";
1034 V = 10:H = 18: GOSUB 500:P =  VAL (B$)
1040  VTAB (11): PRINT "% INTEREST/YEAR = ";
1042 V = 11:H = 18: GOSUB 500:I =  VAL (B$)
1044 I = I / 100
1050  VTAB (12): PRINT "# PAYMENTS/YEAR = ";
1052 V = 12:H = 18: GOSUB 500:N =  VAL (B$)
1055  VTAB (13): PRINT "BUYER'S NAME    = ";
1056 V = 13:H = 18: GOSUB 400:NM$ = B$
1057  VTAB (14): PRINT "SELLER'S NAME   = ";
1058 V = 14:H = 18: GOSUB 400:SN$ = B$
1060  VTAB (15): PRINT "STARTING YEAR-YY= ";
1062 V = 15:H = 18: GOSUB 500:SY% =  VAL (B$)
1065  VTAB (16): PRINT "STARTING MO. -MM= ";
1067 V = 16:H = 18: GOSUB 500:SM% =  VAL (B$)
1070  VTAB (17): PRINT "STARTING DAY -DD= ";
1072 V = 17:H = 18: GOSUB 500:SD% =  VAL (B$)
1075  IF SY% <  = 100 THEN SY% = SY% + 1900
1080 N1% = N * Y + M
1082  IF N = 1 THEN ST% = QQ(1): GOTO 1096
1084  IF N = 2 THEN ST% = QQ(2): GOTO 1096
1086  IF N = 4 THEN ST% = QQ(3): GOTO 1096
1088 ST% = 1
1096 PG% = 1:C1 = 0:I2 = 0:I3 = 0
1098 B0 = P:A1 = 0:A2 = 0
1100  HOME : VTAB (4): PRINT "DO THE FOLLOWING;"
1102  PRINT  TAB( 5)"1- LOAD PRINTER PAPER"
1105  PRINT  TAB( 5)"2- TURN ON PRINTER".
1110  PRINT  TAB( 5)"3- SET TOP-OF-FORM"
1112  PRINT  TAB( 5)"4- PRINTER ON-LINE"
1115  PRINT : INPUT "TYPE RETURN KEY TO
CONTINUE...";ANS$
1120  VTAB (12): CALL  - 958: INPUT "ENTER SLOT(1-7)
";SL
1122  IF SL > 0 AND SL < 8 THEN 1125
1123  VTAB (22): HTAB 10: INVERSE : PRINT "ILLEGAL
ENTRY"; CHR$ (7): NORMAL
```

```
1124  FOR L = 1 TO 1000: NEXT L: GOTO 1120
1125  VTAB (14): CALL  - 958: PRINT "PRINTER TYPE"
1130  PRINT  TAB( 5)"1- SERIAL"
1135  PRINT  TAB( 5)"2- PARALLEL"
1140  PRINT : INPUT "WHICH ONE? ==> ";SP
1142  IF SP = 1 OR SP = 2 THEN 1148
1143  VTAB (22): HTAB 10: INVERSE : PRINT "ILLEGAL
ENTRY"; CHR$ (7): NORMAL
1144  FOR L = 1 TO 1000: NEXT L: GOTO 1125
1148  PR# SL
1150  IF SP = 2 THEN  PRINT C$;"80N"
1200  GOSUB 700
1205  FOR J1 = 1 TO N1%
1210 I1 =  INT ((B0 * I / N) * 100 + 0.5) / 100
1215 C1 = C1 + 1
1220 A = R - I1
1225 A1 = A1 + A
1230 B0 = P - A1
1235  IF C1 <  > N * Y + M THEN 1250
1240 R = R + B0:A = A + B0:A1 = A1 + B0
1245 B0 = 0
1250 I2 = I2 + I1:I3 = I3 + I1:A2 = A2 + A
1255 NR = I1: GOSUB 200: GOSUB 300:D1$ = F$
1260 NR = A: GOSUB 200: GOSUB 300:D2$ = F$
1265 NR = B0: GOSUB 200: GOSUB 300:D3$ = F$
1270 NR = I2: GOSUB 200: GOSUB 300:D4$ = F$
1275  PRINT SM%;"/";SD%;"/";SY%,D1$,D2$,D3$,D4$
1280 K = K + 1: REM   LINE COUNT
1285 D$ = " ": GOSUB 55
1290  IF SM% <  = 12 THEN 1320
1295  GOSUB 800: GOSUB 55
1300 K = K + 3:I3 = 0:A2 = 0
1305  IF (58 - K) >  = 14 THEN 1320
1310  PRINT  CHR$ (12):PG% = PG% + 1
1315  GOSUB 700: REM   HEADER SUBR
1320  NEXT J1
1325  GOSUB 800: GOSUB 900
1330  PRINT " ": PRINT  CHR$ (12)
1335  IF SP = 2 THEN  PRINT C$;"40N"
1340  PR# 0
1345  VTAB 20: INPUT "WANT ANOTHER CALCULATION?(Y/N)
";ANS$
1350  IF  LEFT$ (ANS$,1) = "Y" THEN 1000
1355  VTAB 22: PRINT "PROGRAM CALLED = MAIN MENU"
1360 P$ = "ID.000"
1365  PRINT  CHR$ (4);"RUN ";P$;",D1"
```

圖12-7 攤銷表程式

ID.060 = APR表程式

```
0   POKE 1011,224
1   REM ***** ID.060 ***********
2   REM *
3   REM * WRITTEN BY:JL CAMPBELL
4   REM *        DATED:12/26/80
5   REM *
6   REM * LANGUAGE:FPBASIC
7   REM * COMPUTER:APPLE II
8   REM * APR PROGRAM
9   REM ************************
10   TEXT : CLEAR : POKE 216,0:A =  FRE (0)
12   DIM X(5,3),XI(10),DY(12)
20   FOR J = 1 TO 12: READ DY(J): NEXT J
22   DATA  31,28,31,30,31,30,31,31,30,31,30,31
50   GOTO 950
51   REM **************
52   REM * GEN'L SUBR *
53   REM **************
55   INPUT "WHICH ONE ==> ";N: RETURN
60   VTAB (23): HTAB 13: INVERSE : PRINT "ILLEGAL
ENTRY";: NORMAL : PRINT  CHR$ (7)
64   FOR L = 1 TO 1000: NEXT L: RETURN
70   VTAB (23): INPUT "TYPE RETURN KEY TO CONTINUE...
";Z$
72   VTAB (10): CALL  - 958: RETURN
98   REM * STRING FILTER
100 IX = 1:B$ = "":SW = 0
102  VTAB (V): HTAB H + IX
104  PRINT "+";
106  VTAB (V): HTAB H + IX
108  GET X$
110  IF  ASC (X$) = 13 THEN 128
112  IF  ASC (X$) = 8 THEN 120
113  IF  ASC (X$) = 32 THEN 116
114  IF  ASC (X$) = 46 THEN 116
115  IF  ASC (X$) < 65 OR  ASC (X$) > 90 THEN 106
116 B$ = B$ + X$: PRINT X$
118 IX = IX + 1: GOTO 102
120  IF IX = 1 THEN 102
122  HTAB H + IX: PRINT " ";: HTAB H + IX - 1
124 B$ =  MID$ (B$,1, LEN (B$) - 1)
126 IX = IX - 1: GOTO 102
128  CALL  - 868
130  VTAB (V): HTAB H + 1: PRINT B$
132  RETURN
```

```
198  REM * NUMBER FILTER
200 IX = 1:B$ = "":SW = 0
202  VTAB (V): HTAB H + IX
204  PRINT "+";
206  VTAB (V): HTAB H + IX
208  GET X$
210  IF  ASC (X$) = 13 THEN 232
212  IF  ASC (X$) = 8 THEN 224
213  IF IX > 1 THEN 216
214  IF  ASC (X$) = 45 THEN 220
216  IF  ASC (X$) = 46 AND SW = 0 THEN SW = 1: GOTO 220
218  IF  ASC (X$) < 48 OR  ASC (X$) > 57 THEN 206
220 B$ = B$ + X$: PRINT X$
222 IX = IX + 1: GOTO 202
224  IF IX = 1 THEN 202
226  HTAB H + IX: PRINT " ";: HTAB H + IX - 1
227  IF  MID$ (B$, LEN (B$),1) = "." THEN SW = 0
228 B$ =  MID$ (B$,1, LEN (B$) - 1)
230 IX = IX - 1: GOTO 202
232  CALL  - 868
234  VTAB (V): HTAB H + 1: PRINT B$
236  RETURN
249  REM * SELECT NOTE
250  PRINT "SELECT:"
260  PRINT  TAB( 6)"1- NOTE"
270  PRINT  TAB( 6)"2- NOTE + C.L."
280  PRINT  TAB( 6)"3- NOTE + C.L. + A & H": PRINT
290  RETURN
299  REM * ANNUAL INTEREST
300 I2 = 1:P1 = X(2,1)
305  FOR J = 1 TO 100
310 I1 = (P1 * C / P2) * (1 - (1 + I2 / C) ^ ( - Y *
C))
315  IF  ABS (I2 - I1) <  = 0.000001 THEN 350
320 I2 = I1
325  NEXT J
350 I2 = I2 * 100:I2 =  INT (I2 * 100 + 0.5) / 10000
355  PRINT
360  PRINT "THE ANNUAL INTEREST = ";I2 * 100;" %"
365  PRINT  TAB( 22)"--------"
370  RETURN
400  REM * HEADER SUBR
405  PRINT  TAB( 60)"PAGE ";G
410  PRINT  TAB( 27)"** LOAN SCHEDULE REPORT **": PRINT
TAB( 60)"DATE: ";T$
415  PRINT  TAB( 3)"BUYER: ";N$;: POKE 36,43: PRINT
"SELLER: ";S$
```

```
420  PRINT  TAB( 3)"PRINCIPAL $";P;" AT ";X(3,1) *
100;" % ADD-ON FOR "; INT (Y);" YEARS "; INT (M);"
MONTHS."
425  PRINT  TAB( 3)"REGULAR PAYMENT = ";P1
430  PRINT " "
432  PRINT "DATE";
434  POKE 36,16: PRINT "INTEREST";
436  POKE 36,30: PRINT "PRINCIPAL";
438  POKE 36,47: PRINT "BALANCE";
440  POKE 36,57: PRINT "ACCUM INTEREST"
445  FOR L = 1 TO 78: PRINT "-";: NEXT L: PRINT " "
450 K = 6
460  RETURN
499  REM * YEAR END SUBR
500  PRINT  TAB( 12)"------------  ------------"
502  PRINT "TOTAL ";Y1;" ";
504 NR = K3: GOSUB 800: PRINT F$;"  ";
506 NR = A2: GOSUB 800: PRINT F$
510  PRINT " "
515  PRINT " "
520  RETURN
549  REM * LAST PYMT SUBR
550 NR = P1: GOSUB 800
555  PRINT "LAST PAYMENT = ";F$
560  RETURN
650  PRINT  TAB( 13)"NOTE";
655  HTAB 19: PRINT "+";
660  HTAB 22: PRINT "C. L.";
665  HTAB 29: PRINT "+";
670  HTAB 32: PRINT "A & H"
700  PRINT "# PAYMENTS= "; INT (X(4,1) + 0.5);: HTAB
22: PRINT  INT (X(4,2) + .5);: HTAB 32: PRINT  INT
(X(4,3) + 0.5)
705  PRINT "PRINCIPAL =$";X(1,1);: HTAB 22: PRINT
X(1,2);: HTAB 32: PRINT X(1,3)
710  PRINT "ADD-ON %  = ";X(3,1) * 100;: HTAB 22: PRINT
X(3,2) * 100;: HTAB 32: PRINT X(3,3) * 100
715  PRINT "COMPOUND  = ";"MONTHLY";: HTAB 22: PRINT
"MONTHLY";: HTAB 32: PRINT "MONTHLY"
720  RETURN
750  IF M1 <  = 12 THEN 765
755 M1 = 0
760 Y1 = Y1 + 1
765 M1 = M1 + 1
770  RETURN
799  REM * DECOMPOSE #
800  FOR JX = 1 TO 10:XI(JX) = 0: NEXT JX
805 NR = NR * 100:F$ = ""
```

```
810  FOR JX = 10 TO 1 STEP  - 1
815 X1 =  INT (NR / 10):X2 =  INT (X1 * 10):X3 =  INT
(NR - X2)
820 XI(JX) = X3:NR = X1
825  NEXT JX
830  IF XI(1) = 0 THEN 840
832 F$ =  STR$ (XI(1)) +  STR$ (XI(2)) + ","
833 F$ = F$ +  STR$ (XI(3)) +  STR$ (XI(4)) +  STR$
(XI(5)) + ","
834 F$ = F$ +  STR$ (XI(6)) +  STR$ (XI(7)) +  STR$
(XI(8)) + "."
835 F$ = F$ +  STR$ (XI(9)) +  STR$ (XI(10))
836  RETURN
840  IF XI(2) = 0 THEN 850
842 X$ = " " +  STR$ (XI(2) + ","
844  GOTO 833
850  IF XI(3) = 0 THEN 860
852 F$ = "    "
854  GOTO 833
860  IF XI(4) = 0 THEN 870
862 F$ = "     " +  STR$ (XI(4)) +  STR$ (XI(5)) + ","
864  GOTO 834
870  IF XI(5) = 0 THEN 880
872 F$ = "      " +  STR$ (XI(5)) + ","
874  GOTO 834
880  IF XI(6) = 0 THEN 890
882 F$ = "        "
884  GOTO 834
890  IF XI(7) = 0 THEN 900
892 F$ = "         " +  STR$ (XI(7)) +  STR$ (XI(8)) +
"."
894  GOTO 835
900  IF XI(8) = 0 THEN 910
902 F$ = "          " +  STR$ (XI(8)) + "."
904  GOTO 835
910 F$ = "           "
912  GOTO 835
949  REM * MAIN DRIVER
950  HOME : PRINT : PRINT  TAB( 6)"*** ADD-ON LOAN
SCHEDULES ***": PRINT
960  PRINT "ENTER TODAY'S DATE(MMDDYY) ";
962 V = 4:H = 27: GOSUB 200:T$ = B$
964  IF B$ = "" AND IX <  = 2 THEN 1000
968 L =  LEN (T$): IF L = 6 THEN 974
970 DY(2) = 28: GOSUB 60: GOTO 950
974 Y =  VAL ( MID$ (T$,5,2)):M =  VAL ( LEFT$
(T$,2)):D =  VAL ( MID$ (T$,3,2))
```

```
976  IF  INT (Y / 4) = Y / 4 THEN DY(2) = 29
978  IF M > 0 AND M < 13 THEN 982
980  GOTO 970
982  IF D > 0 AND D <  = DY(M) THEN 990
984  GOTO 970
990 T$ =  LEFT$ (T$,2) + "/" +  MID$ (T$,3,2) + "/" +
MID$ (T$,5,2)
1000  VTAB (4): CALL  - 958:C = 12
1010  VTAB (4): PRINT "ADD-ON INTEREST RATE ==> ";
1012 V = 4:H = 25: GOSUB 200:I =  VAL (B$)
1020 I = I / 100: FOR J = 1 TO 3:X(3,J) = I: NEXT J
1030  VTAB (5): PRINT "NOTE AMOUNT BORROWED ==>$";
1032 V = 5:H = 25: GOSUB 200:X(1,1) =  VAL (B$):P2 =
X(1,1)
1040  VTAB (6): INPUT "# OF YEARS,MONTHS    ==> ";Y,M
1042 Y = (Y * C + M) / C:Y =  INT (Y * 100) / 100
1044 X(4,1) =  INT (Y * C):X(4,2) = X(4,1):X(4,3) =
X(4,1)
1050  VTAB (7): PRINT "PAYMENTS PER YEAR    ==> ";C
1060  VTAB (8): PRINT "CREDIT LIFE INS. AMT.==>$";
1062 V = 8:H = 25: GOSUB 200:A1 =  VAL (B$)
1070  VTAB (9): PRINT "ACCIDENT/HEALTH      ==>$";
1072 V = 9:H = 25: GOSUB 200:A2 =  VAL (B$)
1074 X(1,2) = X(1,1) + A1:X(1,3) = X(1,2) + A2
1080  GOSUB 2000
1100  VTAB (22): CALL  - 958
1102  INPUT "WANT SCHEDULE DISPLAYED?(Y/N) ==> ";A$
1104  IF  LEFT$ (A$,1) = "Y" THEN 4500
1106  IF  LEFT$ (A$,1) = "N" THEN 1110
1108  GOSUB 60: GOTO 1100
1110  VTAB (22): CALL  - 958
1112  INPUT "WANT SCHEDULE PRINTED?(Y/N) ==> ";A$
1114  IF  LEFT$ (A$,1) = "Y" THEN 7000
1116  IF  LEFT$ (A$,1) = "N" THEN 1120
1118  GOSUB 60: GOTO 1110
1120  VTAB 22: CALL  - 958
1122  INPUT "WANT ANOTHER CALCULATION?(Y/N) ==> ";A$
1124  IF  LEFT$ (A$,1) = "Y" THEN 950
1126  IF  LEFT$ (A$,1) = "N" THEN 6000
1128  GOSUB 60: GOTO 1120
2000  REM * COMPUTE PYMT
2020  VTAB (3): CALL  - 958
2030  GOSUB 650
2040 X(2,1) = X(1,1) + X(1,1) * X(3,1) * Y
2042 X(2,1) = X(2,1) / X(4,1)
2050 X(2,1) =  INT (X(2,1) * 100 + 0.5) / 100
2060 X(2,2) = X(1,2) + X(1,2) * X(3,2) * Y
```

```
2062 X(2,2) = X(2,2) / X(4,2)
2070 X(2,2) =  INT (X(2,2) * 100 + 0.5) / 100
2080 X(2,3) = X(1,3) + X(1,3) * X(3,3) * Y
2082 X(2,3) = X(2,3) / X(4,3)
2090 X(2,3) =  INT (X(2,3) * 100 + 0.5) / 100
2100  PRINT : PRINT "PAYMENT   =$";X(2,1);: HTAB 22:
PRINT X(2,2);: HTAB 32: PRINT X(2,3)
2110  FOR K = 1 TO 3: PRINT  TAB( 12 + 10 * (K -
1))"--------";: NEXT K: PRINT
2120  GOSUB 300
2130  FOR K = 1 TO 3:X(5,K) = I2: NEXT K
2995  RETURN
4500  VTAB (11): CALL  - 958
4510  GOSUB 250
4550  VTAB (17): CALL  - 958
4560  GOSUB 55
4570  IF N > 0 AND N < 4 THEN 4600
4580  GOSUB 60: GOTO 4550
4600 P = X(1,N):T1 = 0
4610 P1 = X(2,N)
5000  VTAB (3): CALL  - 958
5005  GOSUB 650
5010  PRINT "PAYMENT   =$";X(2,1);: HTAB 22: PRINT
X(2,2);: HTAB 32: PRINT X(2,3)
5020  PRINT " #";
5022  PRINT  SPC( 3)"BAL.";
5024  PRINT  SPC( 5)"PRIN.";
5026  PRINT  SPC( 4)"INT.";
5028  PRINT  SPC( 4)"T. INT."
5035  FOR J = 1 TO Y * C
5040 K2 = P1 - (I2 * P / C): IF (K2 >  = P) THEN K2 = P
5050 K3 = P1 - K2:P = P - K2: IF P <  = 0 THEN P = 0.0
5052 K4 = P
5055 T1 = T1 + K3
5065  PRINT J;: HTAB 5: PRINT  INT (K4 * 100 + 0.5) /
100;
5070  HTAB 14: PRINT  INT (K2 * 100 + 0.5) / 100;: HTAB
23: PRINT  INT (K3 * 100 + 0.5) / 100;
5075  HTAB 31: PRINT  INT (T1 * 100 + 0.5) / 100
5080  IF  INT (J / C) * C =  INT (J * C) / C THEN
GOSUB 70
5090  NEXT J
5095  GOTO 1110
6000  VTAB 22: CALL  - 958
6010  PRINT "PROGRAM CALLED = MAIN MENU"
6015 P$ = "ID.000"
6020  PRINT  CHR$ (4);"RUN ";P$;",D1"
7000  VTAB (3): CALL  - 958: PRINT
```

```
7010  GOSUB 250
7050  VTAB (9): CALL  - 958
7060  GOSUB 55
7070  IF N > 0 AND N < 4 THEN 7100
7080  GOSUB 60: GOTO 7050
7100  PRINT
7105  VTAB (11): PRINT "ENTER BUYER'S NAME     ==> ";
7107 V = 11:H = 27: GOSUB 100:N$ = B$
7110  VTAB (12): PRINT "ENTER SELLER'S NAME    ==> ";
7112 V = 12:H = 27: GOSUB 100:S$ = B$
7120  VTAB (13): PRINT "ENTER STARTING YEAR -YY==> ";
7122 V = 13:H = 27: GOSUB 200:Y1 =  VAL (B$)
7130  VTAB (14): PRINT "ENTER STARTING MONTH-MM==> ";
7132 V = 14:H = 27: GOSUB 200:M1 =  VAL (B$)
7140  VTAB (15): PRINT "ENTER STARTING DAY  -DD==> ";
7142 V = 15:H = 27: GOSUB 200:D1 =  VAL (B$)
7144 IF D1 > 0 AND D1 < DT(M1) + 1) THEN 7150
7146 GOSUB 60 : GOTO 7140
7150  IF Y1 < 100 THEN Y1 = Y1 + 1900
7160 I = I2:P = X(1,N)
7170 C1 = 0:K3 = 0:K2 = 0
7180 G = 1:P1 = X(2,N)
7190 B0 = P:A1 = 0:A2 = 0
7200  VTAB (4): CALL  - 958
7210  PRINT "DO THE FOLLOWING;"
7215  PRINT  TAB( ]0)"1- LOAD PRINTER PAPER"
7220  PRINT  TAB( 10)"2- TURN ON PRINTER"
7225  PRINT  TAB( 10)"3- SET TOP-OF-FORM"
7230  PRINT  TAB( 10)"4- PRINTER ON-LINE"
7240  PRINT : INPUT "TYPE RETURN KEY TO
CONTINUE....";Z$
7250  VTAB (12): CALL  - 958
7255  INPUT "ENTER PRINTER SLOT(1-7) = ";SL
7260  IF SL > 0 AND SL < 8 THEN 7270
7265  GOSUB 60: GOTO 7250
7270  VTAB (14): CALL  - 958
7275  PRINT "PRINTER TYPE:"
7280  PRINT  TAB( 10)"1- SERIAL"
7285  PRINT  TAB( 10)"2- PARALLEL"
7290  PRINT : INPUT "WHICH ONE? = ";SP
7292  IF SP = 1 OR SP = 2 THEN 7300
7294  GOSUB 60: GOTO 7270
7300  PR# SL
7305  IF SP = 2 THEN  PRINT  CHR$ (9);"80N"
7310  GOSUB 400
```

```
7320  FOR J1 = 1 TO X(4,1)
7330 I1 =  INT ((B0 * I / C) * 100 + 0.5) / 100
7340 C1 = C1 + 1:A = P1 - I1:A1 = A1 + A:B0 = P - A1
7350  IF C1 <  > X(4,1) THEN 7400
7360 P1 = P1 + B0:A = A + B0:A1 = A1 + B0
7370 B0 = 0
7400 K2 = K2 + I1:K3 = K3 + I1:A2 = A2 + A
7402 NR = I1: GOSUB 800:D1$ = F$
7403 NR = A: GOSUB 800:D2$ = F$
7404 NR = B0: GOSUB 800:D3$ = F$
7405 NR = K2: GOSUB 800:D4$ = F$
7410  PRINT M1;"/";D1;"/";Y1;
7412  POKE 36,11: PRINT D1$;"   ";D2$;"   ";D3$;"   ";D4$
7430 K = K + 1: REM * LINE COUNT
7440  GOSUB 750
7450  IF M1 <  = C THEN 7510
7460  GOSUB 500: GOSUB 750
7470 K = K + 4:K3 = 0:A2 = 0
7480  IF (58 - K) >  = 14 THEN 7510
7490  PRINT  CHR$ (12):G = G + 1
7500  GOSUB 400: REM * HDR SUBR
7510  NEXT J1
7520  GOSUB 500: GOSUB 550
7530  PRINT " ": PRINT  CHR$ (12)
7540  IF SP = 2 THEN  PRINT  CHR$ (9);"40N"
7550  PR# 0
7995  GOTO 1120
```

圖12-8 APR表程式

ID.070 = 償還點計算法程式

```
0   POKE 1011,224
1   REM ***** ID.070 ***********
2   REM *
3   REM * WRITTEN BY:JL CAMPBELL
4   REM *       DATED:12/26/80
5   REM *
6   REM * LANGUAGE:FPBASIC
7   REM * COMPUTER:APPLE II
8   REM * PAYBACK METHOD
9   REM ************************
10   TEXT : CLEAR : POKE 216,0:A =  FRE (0)
20   DIM X(3),A(10,3),XI(10),A$(10)
50   GOTO 1000
51   REM **************
52   REM * GEN'L SUBR *
53   REM **************
55   INPUT "WHICH ONE? ==> ";N: RETURN
60   VTAB (23): HTAB 13: INVERSE : PRINT "ILLEGAL
ENTRY";: NORMAL : PRINT  CHR$ (7)
64   FOR L = 1 TO 1000: NEXT L: RETURN
70   INPUT "ANY CHANGES?(Y/N) = ";N$: RETURN
100   VTAB (22): INPUT "TYPE RETURN TO CONTINUE... ";Z$
102   VTAB (9): CALL  - 958: RETURN
210   PRINT "1- INITIAL PROJECT COST  ==>$": RETURN
220   PRINT "2- # OF CASH FLOW PERIODS==> ": RETURN
299   REM  * PRINT USING
300   FOR JX = 1 TO 10:XI(JX) = 0: NEXT JX
305 XD = XD * 100:X$ = ""
310   FOR JX = 10 TO 1 STEP  - 1
315 X1 =  INT (XD / 10):X2 =  INT (X1 * 10):X3 =  INT
(XD - X2)
320 XI(JX) = X3:XD = X1
325   NEXT JX
330   IF XI(1) = 0 THEN 340
332 X$ =  STR$ (XI(1)) +  STR$ (XI(2)) + ","
333 X$ = X$ +  STR$ (XI(3)) +  STR$ (XI(4)) +  STR$
(XI(5)) + ","
334 X$ = X$ +  STR$ (XI(6)) +  STR$ (XI(7)) +  STR$
(XI(8)) + "."
335 X$ = X$ +  STR$ (XI(9)) +  STR$ (XI(10))
336   RETURN
340   IF XI(2) = 0 THEN 350
342 X$ = " " +  STR$ (XI(2)) + ","
344   GOTO 333
```

```
350  IF XI(3) = 0 THEN 360
352 X$ = "    "
354  GOTO 333
360  IF XI(4) = 0 THEN 370
362 X$ = "     " +  STR$ (XI(4)) +  STR$ (XI(5)) + ","
364  GOTO 334
370  IF XI(5) = 0 THEN 380
372 X$ = "      " +  STR$ (XI(5)) + ","
374  GOTO 334
380  IF XI(6) = 0 THEN 390
382 X$ = "         "
384  GOTO 334
390  IF XI(7) = 0 THEN 400
392 X$ = "           " +  STR$ (XI(7)) +  STR$ (XI(8)) +
"."
394  GOTO 335
400  IF XI(8) = 0 THEN 410
402 X$ = "            " +  STR$ (XI(8)) + "."
410 X$ = "             "
412  GOTO 335
499  REM * NUMBER FILTER
500 IX = 1:B$ = "":SW = 0
505  VTAB (V): HTAB H + IX
510  PRINT "+";
515  VTAB (V): HTAB H + IX
520  GET X$
525  IF  ASC (X$) = 13 THEN 580
530  IF  ASC (X$) = 8 THEN 560
532  IF IX > 1 THEN 540
535  IF  ASC (X$) = 45 THEN 550
540  IF  ASC (X$) = 46 AND SW = 0 THEN SW = 1: GOTO 550
545  IF  ASC (X$) < 48 OR  ASC (X$) > 57 THEN 515
550 B$ = B$ + X$: PRINT X$
555 IX = IX + 1: GOTO 505
560  IF IX = 1 THEN 505
565  HTAB H + IX: PRINT " ";: HTAB H + IX - 1
567  IF  MID$ (B$, LEN (B$),1) = "." THEN SW = 0
570 B$ =  MID$ (B$,1, LEN (B$) - 1)
575 IX = IX - 1: GOTO 505
580  CALL  - 868
585  VTAB (V): HTAB H + 1: PRINT B$
590  RETURN
```

```
999  REM  * MAIN DRIVER
1000  HOME : PRINT : PRINT  TAB( 10)"*** PAYBACK METHOD
***": PRINT
1010  FOR J = 1 TO 2: VTAB (3 + J)
1020  ON J GOSUB 210,220
1030  NEXT J
1050  FOR J = 1 TO 2
1060 H = 29:V = 3 + J: GOSUB 500:X(J) =  VAL (B$)
1070  NEXT J
1080  IF X(2) > 0 AND X(2) < 11 THEN 1100
1082  GOSUB 60: VTAB (5): HTAB 30: CALL  - 958
1084 H = 29:V = 5: GOSUB 500:X(2) =  VAL (B$)
1086  GOTO 1080
1100  VTAB (7): CALL  - 958: GOSUB 70
1150  IF  LEFT$ (N$,1) = "N" THEN 2000
1160  IF  LEFT$ (N$,1) = "Y" THEN 1200
1170  GOSUB 60: GOTO 1100
1200  VTAB (7): CALL  - 958
1202  PRINT "0- EXIT CHANGES"
1208  VTAB (9): CALL  - 958
1210  GOSUB 55
1220  IF N = 0 THEN 2000
1230  IF N > 0 AND N < 3 THEN 1250
1240  GOSUB 60: GOTO 1208
1250  VTAB (3 + N): HTAB 30: CALL  - 868
1260 H = 29:V = 3 + N: GOSUB 500:X(N) =  VAL (B$)
1270  IF X(2) > 0 AND X(2) < 11 THEN 1280
1275  GOTO 1080
1280  GOTO 1200
2000  VTAB (7): CALL  - 958
2010  FOR J = 1 TO X(2): VTAB (7 + J)
2020  PRINT "CASH FLOW FOR PERIOD ";J;" ==>$";
2030 H = 28:V = 7 + J: GOSUB 500:A(J,1) =  VAL (B$)
2040  NEXT J
2050  VTAB (9 + X(2)): CALL  - 958
2060  GOSUB 70
2070  IF  LEFT$ (N$,1) = "Y" THEN 2100
2080  IF  LEFT$ (N$,1) = "N" THEN 3000
2090  GOSUB 60: GOTO 2050
2100  VTAB (9 + X(2)): CALL  - 958
2110  PRINT "0- EXIT CHANGES"
2120  PRINT : GOSUB 55
2130  IF N = 0 THEN 3000
2140  IF N > 0 AND N < X(2) + 1 THEN 2200
2150  GOSUB 60: GOTO 2100
2200  PRINT : PRINT "CASH FLOW FOR PERIOD ==>$";
2205 H = 26:V = 13 + X(2): GOSUB 500:A(N,1) =  VAL (B$)
```

```
2210  VTAB (8): CALL  - 958
2220  FOR J = 1 TO X(2)
2230  PRINT "CASH FLOW FOR PERIOD ";J;" ==>$";A(J,1)
2240  NEXT J
2250  GOTO 2050
3000  VTAB (7): CALL  - 958
3010  PRINT : PRINT  TAB( 10)".....PROCESSING....."
3020  PRINT : PRINT  TAB( 10)"..I'M WORKING HARD.."
3025 T = X(1)
3030  FOR J = 1 TO X(2)
3040 T = T - A(J,1)
3050  IF T < 0 THEN T1 = (T / A(J,1)) + J: GOTO 3100
3060  NEXT J
3100  VTAB (7): CALL  - 958
3110  FOR J = 1 TO X(2):XD =  ABS (A(J,1)): GOSUB 300
3115  IF A(J,1) < 0 THEN X$ = X$ + "-"
3120  PRINT "CASH FLOW - PERIOD ";J;" ==>$";X$:A$(J) =
X$
3130  NEXT J
3140  PRINT
3150  PRINT "PAYBACK PERIOD EQUALS ==> "; INT (T1 * 100
+ 0.5) / 100
3200  VTAB (20): CALL  - 958: INPUT "WANT PRINTED
REPORT?(Y/N) ";N$
3205  IF  LEFT$ (N$,1) = "Y" THEN 3220
3210  IF  LEFT$ (N$,1) = "N" THEN 3500
3215  GOSUB 60: GOTO 3200
3220  VTAB (4): CALL  - 958: PRINT "DO THE FOLLOWING:"
3225  PRINT  TAB( 5)"1- LOAD PRINTER PAPER"
3230  PRINT  TAB( 5)"2- TURN ON PRINTER"
3235  PRINT  TAB( 5)"3- SET TOP-OF-FORM"
3240  PRINT  TAB( 5)"4- PRINTER ON-LINE"
3245  PRINT : INPUT "TYPE RETURN KEY TO
CONTINUE....";N$
3250  VTAB (12): CALL  - 958: INPUT "PRINTER SLOT(1-7)
";SL
3255  IF SL > 0 AND SL < 8 THEN 3265
3260  GOSUB 60: GOTO 3220
3265  VTAB (14): CALL  - 958: PRINT "PRINTER TYPE:"
3270  PRINT  TAB( 5)"1- SERIAL"
3275  PRINT  TAB( 5)"2- PARALLEL"
3280  PRINT : GOSUB 55:SP = N
3285  IF SP > 0 AND SP < 3 THEN 3295
3290  GOSUB 60: GOTO 3265
3295  PR# SL
3300  IF SP = 2 THEN  PRINT  CHR$ (9);"80N"
```

```
3305  PRINT  TAB( 29)"PAYBACK METHOD REPORT"
3310  PRINT " "
3315  PRINT  TAB( 10)"1- INITIAL PROJECT COST
==>$";X(1)
3320  PRINT  TAB( 10)"2- # OF CASH FLOW PERIODS==>
";X(2)
3325  PRINT " "
3330  FOR J = 1 TO X(2)
3335  PRINT  TAB( 10)"CASH FLOW - PERIOD ";J;"
==>$";A$(J)
3340  NEXT J
3345  PRINT " "
3350  PRINT  TAB( 10)"PAYBACK PERIOD EQUALS==> "; INT
(T1 * 100 + 0.5) / 100
3355  PRINT  CHR$ (12)
3360  IF SP = 2 THEN  PRINT  CHR$ (9);"40N"
3365  PR# 0
3500  VTAB (20): CALL  - 958: INPUT "WANT ANOTHER
CALCULATION?(Y/N) ";N$
3510  IF  LEFT$ (N$,1) = "Y" THEN 1000
9000  PRINT : PRINT "PROGRAM CALLED =MAIN MENU"
9005 P$ = "ID.000"
9010  PRINT  CHR$ (4);"RUN ";P$;",D1"
```

圖12-9 償還點計算法程式

ID.080 = 淨現值法（NPV）程式

```
0   POKE 1011,224
1   REM ***** ID.080 ***********
2   REM *
4   REM *       DATED:12/26/80
5   REM *
6   REM * LANGUAGE:FPBASIC
7   REM * COMPUTER:APPLE II
8   REM * NPV PROGRAM
9   REM ************************
10   TEXT : CLEAR : POKE 216,0:A =  FRE (0)
20   DIM X(3),A(10,3),XI(10),A$(10)
50   GOTO 1000
51   REM **************
52   REM * GEN'L SUBR *
53   REM **************
55   INPUT "WHICH ONE? ==> ";N: RETURN
60   VTAB (23): HTAB 13: INVERSE : PRINT "ILLEGAL
ENTRY";: NORMAL : PRINT  CHR$ (7)
64   FOR L = 1 TO 1000: NEXT L: RETURN
70   INPUT "ANY CHANGES(Y/N) = ";N$: RETURN
210   PRINT "1- INITIAL PROJECT COST  ==>$": RETURN
220   PRINT "2- # OF CASH FLOW PERIODS==> ": RETURN
230   PRINT "3- INTEREST RATE (%)     ==> ": RETURN
299   REM  * PRINT USING
300   FOR JX = 1 TO 10:XI(JX) = 0: NEXT JX
305 XD = XD * 100:X$ = ""
310   FOR JX = 10 TO 1 STEP  - 1
315 X1 =  INT (XD / 10):X2 =  INT (X1 * 10):X3 =  INT
(XD - X2)
320 XI(JX) = X3:XD = X1
325   NEXT JX
330   IF XI(1) = 0 THEN 340
332 X$ =  STR$ (XI(1)) +  STR$ (XI(2)) + ","
333 X$ = X$ +  STR$ (XI(3)) +  STR$ (XI(4)) +  STR$
(XI(5)) + ","
334 X$ = X$ +  STR$ (XI(6)) +  STR$ (XI(7)) +  STR$
(XI(8)) + "."
335 X$ = X$ +  STR$ (XI(9)) +  STR$ (XI(10))
336   RETURN
340   IF XI(2) = 0 THEN 350
342 X$ = " " +  STR$ (XI(2)) + ","
344   GOTO 333
```

```
350  IF XI(3) = 0 THEN 360
352 X$ = "    "
354  GOTO 333
360  IF XI(4) = 0 THEN 370
362 X$ = "     " +  STR$ (XI(4)) +  STR$ (XI(5)) + ","
364  GOTO 334
370  IF XI(5) = 0 THEN 380
372 X$ = "      " +  STR$ (XI(5)) + ","
374  GOTO 334
380  IF XI(6) = 0 THEN 390
382 X$ = "        "
384  GOTO 334
390  IF XI(7) = 0 THEN 400
392 X$ = "         " +  STR$ (XI(7)) +  STR$ (XI(8)) +
"."
394  GOTO 335
400  IF XI(8) = 0 THEN 410
402 X$ = "          " +  STR$ (XI(8)) + "."
410 X$ = "           "
412  GOTO 335
499  REM * NUMBER FILTER
500 IX = 1:B$ = "":SW = 0
505  VTAB (V): HTAB H + IX
510  PRINT "+";
515  VTAB (V): HTAB H + IX
520  GET X$
525  IF  ASC (X$) = 13 THEN 580
530  IF  ASC (X$) = 8 THEN 560
532  IF IX > 1 THEN 540
535  IF  ASC (X$) = 45 THEN 550
540  IF  ASC (X$) = 46 AND SW = 0 THEN SW = 1: GOTO 55
545  IF  ASC (X$) < 48 OR  ASC (X$) > 57 THEN 515
550 B$ = B$ + X$: PRINT X$
555 IX = IX + 1: GOTO 505
560  IF IX = 1 THEN 505
565  HTAB H + IX: PRINT " ";: HTAB H + IX - 1
567  IF  MID$ (B$, LEN (B$),1) = "." THEN SW = 0
570 B$ =  MID$ (B$,1, LEN (B$) - 1)
575 IX = IX - 1: GOTO 505
580  CALL  - 868
585  VTAB (V): HTAB H + 1: PRINT B$
590  RETURN
```

```
999  REM  * MAIN DRIVER
1000  HOME : PRINT : PRINT  TAB( 8)"*** NET PRESENT
VALUE ***": PRINT
1010  FOR J = 1 TO 3: VTAB (3 + J)
1020  ON J GOSUB 210,220,230
1030  NEXT J
1050  FOR J = 1 TO 3
1060 H = 29:V = 3 + J: GOSUB 500:X(J) =  VAL (B$)
1070  NEXT J
1075 X(3) = X(3) / 100
1080  IF X(2) > 0 AND X(2) < 11 THEN 1100
1082  GOSUB 60: VTAB (5): HTAB 30: CALL  - 868
1084 V = 5:H = 29: GOSUB 500:X(2) =  VAL (B$)
1086  GOTO 1080
1100  VTAB (8): CALL  - 958: GOSUB 70
1150  IF  LEFT$ (N$,1) = "N" THEN 2000
1160  IF  LEFT$ (N$,1) = "Y" THEN 1200
1170  GOSUB 60: GOTO 1100
1200  VTAB (8): CALL  - 958
1202  PRINT "0- EXIT CHANGES"
1208  VTAB (10): CALL  - 958
1210  GOSUB 55
1220  IF N = 0 THEN 2000
1230  IF N > 0 AND N < 4 THEN 1250
1240  GOSUB 60: GOTO 1208
1250  VTAB (3 + N): HTAB 30: CALL  - 868
1260 H = 29:V = 3 + N: GOSUB 500:X(N) =  VAL (B$)
1265  IF N = 3 THEN X(3) = X(3) / 100
1270  IF X(2) > 0 AND X(2) < 11 THEN 1280
1275  GOTO 1082
1280  GOTO 1200
2000  VTAB (8): CALL  - 958
2010  FOR J = 1 TO X(2): VTAB (7 + J)
2020  PRINT "CASH FLOW FOR PERIOD ";J;" ==>$";
2030 H = 28:V = 7 + J: GOSUB 500:A(J,1) =  VAL (B$)
2040  NEXT J
2050  VTAB (9 + X(2)): CALL  - 958
2060  GOSUB 70
2070  IF  LEFT$ (N$,1) = "Y" THEN 2100
2080  IF  LEFT$ (N$,1) = "N" THEN 3000
2090  GOSUB 60: GOTO 2050
2100  VTAB (9 + X(2)): CALL  - 958
2110  PRINT "0- EXIT CHANGES"
2120  PRINT : GOSUB 55
2130  IF N = 0 THEN 3000
2140  IF N > 0 AND N < X(2) + 1 THEN 2200
2150  GOSUB 60: GOTO 2100
2200  PRINT : PRINT "CASH FLOW FOR PERIOD ==>$";
2205 H = 26:V = 13 + X(2): GOSUB 500:A(N,1) =  VAL (B$)
```

```
2210  VTAB (8): CALL  - 958
2220  FOR J = 1 TO X(2)
2230  PRINT "CASH FLOW FOR PERIOD ";J;" ==>$";A(J,1)
2240  NEXT J
2250  GOTO 2050
3000  VTAB (7): CALL  - 958
3010  PRINT : PRINT  TAB( 10)".....PROCESSING....."
3020  PRINT : PRINT  TAB( 10)"..I'M WORKING HARD.."
3025 T = 0
3030  FOR J = 1 TO X(2)
3040 A(J,2) = (1 + X(3)) ^ J
3050 A(J,3) = A(J,1) / A(J,2)
3060 T = T + A(J,3)
3070  NEXT J
3100  VTAB (8): CALL  - 958
3110  FOR J = 1 TO X(2):XD =  ABS (A(J,3)): GOSUB 300
3115  IF A(J,3) < 0 THEN X$ = X$ + "-"
3120  PRINT "CASH FLOW - PERIOD ";J;" ==>$";X$:A$(J) =
X$
3130  NEXT J
3140  PRINT
3145 XD = T: GOSUB 300:T1$ = X$
3150  PRINT "TOTAL FOR ALL PERIODS==>$";X$
3155 XD = X(1): GOSUB 300:T2$ = X$
3160  PRINT "INITIAL PROJECT COST ==>$";X$
3170  PRINT  TAB( 26)"-------------"
3180 T1 = T - X(1)
3190  IF T1 < 0 THEN 3200
3192 XD = T1: GOSUB 300:T3$ = X$
3195  PRINT "NET PRESENT VALUE    ==>$";X$
3197  GOTO 3250
3200 T1 =  ABS (T1):XD = T1: GOSUB 300
3210  PRINT "NPV IS NEGATIVE-ABORT==>-";X$
3250  VTAB (20): CALL  - 958: INPUT "WANT PRINTED
REPORT?(Y/N) ==> ";N$
3255  IF  LEFT$ (N$,1) = "Y" THEN 3270
3260  IF  LEFT$ (N$,1) = "N" THEN 3500
3265  GOSUB 60: GOTO 3250
3270  VTAB (4): CALL  - 958: PRINT "DO THE FOLLOWING:"
3275  PRINT  TAB( 5)"1- LOAD PRINTER PAPER"
3280  PRINT  TAB( 5)"2- TURN ON PRINTER"
3285  PRINT  TAB( 5)"3- SET TOP-OF-FORM"
3290  PRINT  TAB( 5)"4- PRINTER ON-LINE"
3295  PRINT : INPUT "TYPE RETURN KEY TO
CONTINUE....";N$
```

```
3300  VTAB (12): CALL  - 958: INPUT "PRINTER SLOT(1-7)
";SL
3305  IF SL > 0 AND SL < 8 THEN 3315
3310  GOSUB 60: GOTO 3300
3315  VTAB (14): CALL  - 958: PRINT "PRINTER TYPE:"
3320  PRINT  TAB( 5)"1- SERIAL"
3325  PRINT  TAB( 5)"2- PARALLEL"
3330  PRINT : GOSUB 55:SP = N
3335  IF SP > 0 AND SP < 3 THEN 3345
3340  GOSUB 60: GOTO 3315
3345  PR# SL
3350  IF SP = 2 THEN  PRINT  CHR$ (9);"80N"
3355  PRINT  TAB( 25)"NPV - NET PRESENT VALUE REPORT"
3360  PRINT " "
3365  PRINT  TAB( 10)"1- INITIAL PROJECT COST
==>$";X(1)
3370  PRINT  TAB( 10)"2- # OF CASH FLOW
PERIODS==>$";X(2)
3375  PRINT  TAB( 10)"3- INTEREST RATE (%)     ==> ";
INT (X(3) * 10000 + 0.5) / 100
3380  PRINT " "
3385  FOR J = 1 TO X(2)
3390  PRINT  TAB( 10)"CASH FLOW - PERIOD ";J;"
==>$";A$(J)
3395  NEXT J
3400  PRINT " "
3405  PRINT  TAB( 10)"TOTAL FOR ALL PERIODS==>$";T1$
3410  PRINT  TAB( 10)"INITIAL PROJECT COST ==>$";T2$
3415  PRINT  TAB( 36)"-------------"
3420  IF T1 < 0 THEN 3450
3425  PRINT  TAB( 10)"NET PRESENT VALUE    ==>$";T3$
3430  GOTO 3480
3450  PRINT  TAB( 10)"NPV IS
NEGATIVE-ABORT==>$";T3$;"-"
3480  PRINT  CHR$ (12)
3490  IF SL = 2 THEN  PRINT  CHR$ (9);"80N"
3495  PR# 0
3500  VTAB (20): CALL  - 958: INPUT "WANT ANOTHER
CALCULATION?(Y/N) ";N$
3510  IF  LEFT$ (N$,1) = "Y" THEN 1000
9000  PRINT : PRINT "PROGRAM CALL = MAIN MENU"
9005 P$ = "ID.000"
9010  PRINT  CHR$ (4);"RUN ";P$;",D1"
```

圖12-10 淨現值法(NPV)程式

ID.090 = IRR 程式

```
0   POKE 1011,224
1   REM ***** ID.090 ***********
2   REM *
3   REM * WRITTEN BY:JL CAMPBELL
4   REM *       DATED:12/26/80
5   REM *
6   REM * LANGUAGE:FPBASIC
7   REM * COMPUTER:APPLE II
8   REM * IRR PROGRAM
9   REM ************************
10   TEXT : CLEAR : POKE 216,0:A =  FRE (0)
20   DIM X(3),A(10,3),XI(10),A$(10)
50   GOTO 1000
51   REM **************
52   REM * GEN'L SUBR *
53   REM **************
55   INPUT "WHICH ONE? ==> ";N: RETURN
60   VTAB (23): HTAB 13: INVERSE : PRINT "ILLEGAL
ENTRY";: NORMAL : PRINT  CHR$ (7)
64   FOR L = 1 TO 1000: NEXT L: RETURN
70   INPUT "ANY CHANGES(Y/N) = ";N$: RETURN
210   PRINT "1- INITIAL PROJECT COST   ==>$": RETURN
220   PRINT "2- # OF CASH FLOW PERIODS==> ": RETURN
299   REM  * PRINT USING
300   FOR JX = 1 TO 10:XI(JX) = 0: NEXT JX
305 XD = XD * 100:X$ = ""
310   FOR JX = 10 TO 1 STEP  - 1
315 X1 =  INT (XD / 10):X2 =  INT (X1 * 10):X3 =  INT
(XD - X2)
320 XI(JX) = X3:XD = X1
325   NEXT JX
330   IF XI(1) = 0 THEN 340
332 X$ =  STR$ (XI(1)) +  STR$ (XI(2)) + ","
333 X$ = X$ +  STR$ (XI(3)) +  STR$ (XI(4)) +  STR$
(XI(5)) + ","
334 X$ = X$ +  STR$ (XI(6)) +  STR$ (XI(7)) +  STR$
(XI(8)) + "."
335 X$ = X$ +  STR$ (XI(9)) +  STR$ (XI(10))
336   RETURN
340   IF XI(2) = 0 THEN 350
342 X$ = ". " +  STR$ (XI(2)) + ","
344   GOTO 333
```

```
350  IF XI(3) = 0 THEN 360
352 X$ = "    "
354  GOTO 333
360  IF XI(4) = 0 THEN 370
362 X$ = "     " +  STR$ (XI(4)) +  STR$ (XI(5)) + ","
364  GOTO 334
370  IF XI(5) = 0 THEN 380
372 X$ = "      " +  STR$ (XI(5)) + ","
374  GOTO 334
380  IF XI(6) = 0 THEN 390
382 X$ = "         "
384  GOTO 334
390  IF XI(7) = 0 THEN 400
392 X$ = "         " +  STR$ (XI(7)) +  STR$ (XI(8)) +
"."
394  GOTO 335
400  IF XI(8) = 0 THEN 410
402 X$ = "          " +  STR$ (XI(8)) + "."
410 X$ = "            "
412  GOTO 335
499  REM * NUMBER FILTER
500 IX = 1:B$ = "":SW = 0
505  VTAB (V): HTAB H + IX
510  PRINT "+";
515  VTAB (V): HTAB H + IX
520  GET X$
525  IF  ASC (X$) = 13 THEN 580
530  IF  ASC (X$) = 8 THEN 560
532  IF IX > 1 THEN 540
535  IF  ASC (X$) = 45 THEN 550
540  IF  ASC (X$) = 46 AND SW = 0 THEN SW = 1: GOTO 550
545  IF  ASC (X$) < 48 OR  ASC (X$) > 57 THEN 515
550 B$ = B$ + X$: PRINT X$
555 IX = IX + 1: GOTO 505
560  IF IX = 1 THEN 505
565  HTAB H + IX: PRINT " ";: HTAB H + IX - 1
567  IF  MID$ (B$, LEN (B$),1) = "." THEN SW = 0
570 B$ =  MID$ (B$,1, LEN (B$) - 1)
575 IX = IX - 1: GOTO 505
580  CALL  - 868
585  VTAB (V): HTAB H + 1: PRINT B$
590  RETURN
```

```
999  REM  * MAIN DRIVER
1000  HOME : PRINT : PRINT  TAB( 4)"*** INTERNAL RATE
OF RETURN ***": PRINT
1010  FOR J = 1 TO 2: VTAB (3 + J)
1020  ON J GOSUB 210,220
1030  NEXT J
1050  FOR J = 1 TO 2: VTAB (3 + J): HTAB 30
1060 H = 29:V = 3 + J: GOSUB 500:X(J) =  VAL (B$)
1070  NEXT J
1080  IF X(2) > 0 AND X(2) < 11 THEN 1100
1082  GOSUB 60: VTAB (5): HTAB 30: CALL  - 958
1084 V = 5:H = 29: GOSUB 500:X(2) =  VAL (B$)
1086  GOTO 1080
1100  VTAB (7): CALL  - 958: GOSUB 70
1150  IF  LEFT$ (N$,1) = "N" THEN 2000
1160  IF  LEFT$ (N$,1) = "Y" THEN 1200
1170  GOSUB 60: GOTO 1100
1200  VTAB (7): CALL  - 958
1202  PRINT "0- EXIT CHANGES"
1208  VTAB (9): CALL  - 958
1210  GOSUB 55
1220  IF N = 0 THEN 2000
1230  IF N > 0 AND N < 3 THEN 1250
1240  GOSUB 60: GOTO 1208
1250  VTAB (3 + N): HTAB 30: CALL  - 868
1260 V = 3 + N:H = 29: GOSUB 500:X(N) =  VAL (B$)
1270  IF X(2) > 0 AND X(2) < 11 THEN 1280
1275  GOTO 1082
1280  GOTO 1200
2000  VTAB (7): CALL  - 958
2010  FOR J = 1 TO X(2): VTAB (7 + J)
2020  PRINT "CASH FLOW FOR PERIOD ";J;" ==>$";
2030 H = 28:V = 7 + J: GOSUB 500:A(J,1) =  VAL (B$)
2040  NEXT J
2050  VTAB (9 + X(2)): CALL  - 958
2060  GOSUB 70
2070  IF  LEFT$ (N$,1) = "Y" THEN 2100
2080  IF  LEFT$ (N$,1) = "N" THEN 3000
2090  GOSUB 60: GOTO 2050
2100  VTAB (9 + X(2)): CALL  - 958
2110  PRINT "0- EXIT CHANGES"
2120  PRINT : GOSUB 55
2130  IF N = 0 THEN 3000
2140  IF N > 0 AND N < X(2) + 1 THEN 2200
2150  GOSUB 60: GOTO 2100
2200  PRINT : PRINT "CASH FLOW FOR PERIOD ==>$";
2205 V = 13 + X(2):H = 26: GOSUB 500:A(N,1) =  VAL (B$)
```

```
2210  VTAB (8): CALL  - 958
2220  FOR J = 1 TO X(2)
2230  PRINT "CASH FLOW FOR PERIOD ";J;" ==>$";A(J,1)
2240  NEXT J
2250  GOTO 2050
3000  VTAB (7): CALL  - 958
3010  PRINT : PRINT  TAB( 10)".....PROCESSING....."
3020  PRINT : PRINT  TAB( 10)"..I'M WORKING HARD.."
3025 L = 0:H = 1:T = X(1)
3030 I = (H + L) / 2
3035 T1 = 0
3040  FOR J = 1 TO X(2)
3045 A(J,2) = A(J,1) / ((1 + I) ^ J)
3050 T1 = T1 + A(J,2)
3055  NEXT J
3060  IF  ABS (T - T1) < 10 THEN I1 = I: GOTO 3100
3065  IF T > T1 THEN H = I: GOTO 3075
3070 L = I
3075  VTAB (14): CALL  - 958
3080  PRINT "IRR % = "; INT (I * 10000 + 0.5) / 100
3085  PRINT : PRINT "VALUE = "; INT (T1 * 100 + 0.5) /
100
3090  IF I <  = 0.0001 THEN 3100
3095  GOTO 3030
3100  VTAB (8): CALL  - 958
3110  FOR J = 1 TO X(2):XD =  ABS (A(J,1)): GOSUB 300
3115  IF A(J,1) < 0 THEN X$ = X$ + "-"
3120  PRINT "CASH FLOW - PERIOD ";J;" ==>$";X$:A$(J) =
X$
3130  NEXT J
3140  PRINT
3150  PRINT "INTERNAL RATE EQUALS ==> "; INT (I1 *
10000 + 0.5) / 100;" %"
3200  VTAB (20): CALL  - 958: INPUT "WANT PRINTED
REPORT?(Y/N) ==> ";N$
3205  IF  LEFT$ (N$,1) = "Y" THEN 3220
3210  IF  LEFT$ (N$,1) = "N" THEN 3500
3215  GOSUB 60: GOTO 3200
3220  VTAB (4): CALL  - 958
3225  PRINT "DO THE FOLLOWING:"
3230  PRINT  TAB( 5)"1- LOAD PRINTER PAPER"
3235  PRINT  TAB( 5)"2- TURN ON PRINTER"
3240  PRINT  TAB( 5)"3- SET TOP-OF-FORM"
3245  PRINT  TAB( 5)"4- PRINTER ON-LINE"
3250  PRINT : INPUT "TYPE RETURN KEY TO
CONTINUE....";N$
3255  VTAB (12): CALL  - 958: INPUT "PRINTER SLOT(1-7)
";SL
```

```
3260  IF SL > 0 AND SL < 8 THEN 3270
3265  GOSUB 60: GOTO 3255
3270  VTAB (14): CALL  - 958: PRINT "PRINTER TYPE:"
3275  PRINT  TAB( 5)"1- SERIAL"
3280  PRINT  TAB( 5)"2- PARALLEL"
3285  PRINT : GOSUB 55:SP = N
3290  IF SP0 AND SP < 3 THEN 3300
3295  GOSUB 60: GOTO 3270
3300  PR# SL
3305  IF SP = 2 THEN  PRINT  CHR$ (9);"80N"
3310  PRINT  TAB( 22)"IRR - INTERNAL RATE OF RETURN
REPORT"
3315  PRINT " "
3320  PRINT  TAB( 10)"1- INITIAL PROJECT COST
==>$";X(1)
3325  PRINT  TAB( 10)"2- # OF CASH FLOW PERIODS==>
";X(2)
3335  PRINT " "
3340  FOR J = 1 TO X(2)
3345  PRINT  TAB( 10)"CASH FLOW - PERIOD ";J;"
==>$";A$(J)
3350  NEXT J
3355  PRINT " "
3360  PRINT  TAB( 10)"INTERNAL RATE EQUALS ==> "; INT
(I1 * 10000 + 0.5) / 100;"%"
3370  PRINT  CHR$ (12)
3380  IF SP = 2 THEN  PRINT  CHR$ (9);"40N"
3390  PR# 0
3500  VTAB (20): CALL  - 958: INPUT "WANT ANOTHER
CALCULATION?(Y/N) ";N$
3510  IF  LEFT$ (N$,1) = "Y" THEN 1000
9000  PRINT : PRINT "PROGRAM CALLED = MAIN MENU"
9005 P$ = "ID.000"
9010  PRINT  CHR$ (4);"RUN ";P$;",D1"
```

圖12-11 IRR程式

ID.100 = 獲利率程式

```
0   POKE 1011,224
1   REM ***** ID.100 ***********
2   REM *
3   REM * WRITTEN BY:JL CAMPBELL
4   REM *      DATED:12/26/80
5   REM *
6   REM * LANGUAGE:FPBASIC
7   REM * COMPUTER:APPLE II
8   REM * PROFITABILITY INDEX
9   REM ***********************
10  TEXT : CLEAR : POKE 216,0:A =  FRE (0)
20  DIM X(3),A(10,3),XI(10),A$(10)
50  GOTO 1000
51  REM *************
52  REM * GEN'L SUBR *
53  REM *************
55  INPUT "WHICH ONE? ==> ";N: RETURN
60  VTAB (23): HTAB 13: INVERSE : PRINT "ILLEGAL
ENTRY";: NORMAL : PRINT  CHR$ (7)
64  FOR L = 1 TO 1000: NEXT L: RETURN
70  INPUT "ANY CHANGES(Y/N) = ";N$: RETURN
210  PRINT "1- INITIAL PROJECT COST   ==>$": RETURN
220  PRINT "2- # OF CASH FLOW PERIODS==> ": RETURN
230  PRINT "3- INTEREST RATE (%)      ==> ": RETURN
299  REM  * PRINT USING
300  FOR JX = 1 TO 10:XI(JX) = 0: NEXT JX
305 XD = XD * 100:X$ = ""
310  FOR JX = 10 TO 1 STEP  - 1
315 X1 =  INT (XD / 10):X2 =  INT (X1 * 10):X3 =  INT
(XD - X2)
320 XI(JX) = X3:XD = X1
325  NEXT JX
330  IF XI(1) = 0 THEN 340
332 X$ =  STR$ (XI(1)) +  STR$ (XI(2)) + ","
333 X$ = X$ +  STR$ (XI(3)) +  STR$ (XI(4)) +  STR$
(XI(5)) + ","
334 X$ = X$ +  STR$ (XI(6)) +  STR$ (XI(7)) +  STR$
(XI(8)) + "."
335 X$ = X$ +  STR$ (XI(9)) +  STR$ (XI(10))
336  RETURN
340  IF XI(2) = 0 THEN 350
342 X$ = " " +  STR$ (XI(2)) + ","
344  GOTO 333
```

```
350  IF XI(3) = 0 THEN 360
352 X$ = "    "
354  GOTO 333
360  IF XI(4) = 0 THEN 370
362 X$ = "      " +  STR$ (XI(4)) +  STR$ (XI(5)) + ","
364  GOTO 334
370  IF XI(5) = 0 THEN 380
372 X$ = "       " +  STR$ (XI(5)) + ","
374  GOTO 334
380  IF XI(6) = 0 THEN 390
382 X$ = "         "
384  GOTO 334
390  IF XI(7) = 0 THEN 400
392 X$ = "          " +  STR$ (XI(7)) +  STR$ (XI(8)) +
"."
394  GOTO 335
400  IF XI(8) = 0 THEN 410
402 X$ = "           " +  STR$ (XI(8)) + "."
410 X$ = "            "
412  GOTO 335
499  REM * NUMBER FILTER
500 IX = 1:B$ = "":SW = 0
505  VTAB (V): HTAB H + IX
510  PRINT "+";
515  VTAB (V): HTAB H + IX
520  GET X$
525  IF  ASC (X$) = 13 THEN 580
530  IF  ASC (X$) = 8 THEN 560
532  IF IX > 1 THEN 540
535  IF  ASC (X$) = 45 THEN 550
540  IF  ASC (X$) = 46 AND SW = 0 THEN SW = 1: GOTO 550
545  IF  ASC (X$) < 48 OR  ASC (X$) > 57 THEN 515
550 B$ = B$ + X$: PRINT X$
555 IX = IX + 1: GOTO 505
560  IF IX = 1 THEN 505
565  HTAB H + IX: PRINT " ";: HTAB H + IX - 1
567  IF  MID$ (B$, LEN (B$),1) = "." THEN SW = 0
570 B$ =  MID$ (B$,1, LEN (B$) - 1)
575 IX = IX - 1: GOTO 505
580  CALL  - 868
585  VTAB (V): HTAB H + 1: PRINT B$
590  RETURN
```

```
999  REM  * MAIN DRIVER
1000 HOME : PRINT : PRINT  TAB( 7)"*** PROFITABILITY
INDEX ***": PRINT
1010  FOR J = 1 TO 3: VTAB (3 + J)
1020  ON J GOSUB 210,220,230
1030  NEXT J
1050  FOR J = 1 TO 3: VTAB (3 + J): HTAB 29
1060 H = 29:V = 3 + J: GOSUB 500:X(J) =  VAL (B$)
1070  NEXT J
1075 X(3) = X(3) / 100
1080  IF X(2) > 0 AND X(2) < 11 THEN 1100
1082  GOSUB 60: VTAB (5): HTAB 29: CALL  - 868
1084 V = 5:H = 29: GOSUB 500:X(2) =  VAL (B$)
1086  GOTO 1080
1100  VTAB (8): CALL  - 958: GOSUB 70
1150  IF  LEFT$ (N$,1) = "N" THEN 2000
1160  IF  LEFT$ (N$,1) = "Y" THEN 1200
1170  GOSUB 60: GOTO 1100
1200  VTAB (8): CALL  - 958
1202  PRINT "0- EXIT CHANGES"
1208  VTAB (10): CALL  - 958
1210  GOSUB 55
1220  IF N = 0 THEN 2000
1230  IF N > 0 AND N < 4 THEN 1250
1240  GOSUB 60: GOTO 1208
1250  VTAB (3 + N): HTAB 30: CALL  - 868
1260 H = 29:V = 3 + N: GOSUB 500:X(N) =  VAL (B$)
1265  IF N = 3 THEN X(3) = X(3) / 100
1270  IF X(2) > 0 AND X(2) < 11 THEN 1280
1275  GOTO 1082
1280  GOTO 1200
2000  VTAB (8): CALL  - 958
2010  FOR J = 1 TO X(2): VTAB (7 + J)
2020  PRINT "CASH FLOW FOR PERIOD ";J;" ==>$";
2030 H = 28:V = 7 + J: GOSUB 500:A(J,1) =  VAL (B$)
2040  NEXT J
2050  VTAB (9 + X(2)): CALL  - 958
2060  GOSUB 70
2070  IF  LEFT$ (N$,1) = "Y" THEN 2100
2080  IF  LEFT$ (N$,1) = "N" THEN 3000
2090  GOSUB 60: GOTO 2050
2100  VTAB (9 + X(2)): CALL  - 958
2110  PRINT "0- EXIT CHANGES"
2120  PRINT : GOSUB 55
2130  IF N = 0 THEN 3000
2140  IF N > 0 AND N < X(2) + 1 THEN 2200
2150  GOSUB 60: GOTO 2100
2200  PRINT : PRINT "CASH FLOW FOR PERIOD ==>$";
2205 V = 13 + X(2):H = 26: GOSUB 500:A(N,1) =  VAL (B$
```

```
2210  VTAB (8): CALL  - 958
2220  FOR J = 1 TO X(2)
2230  PRINT "CASH FLOW FOR PERIOD ";J;" ==>$";A(J,1)
2240  NEXT J
2250  GOTO 2050
3000  VTAB (7): CALL  - 958
3010  PRINT : PRINT  TAB( 10)".....PROCESSING....."
3020  PRINT : PRINT  TAB( 10)"..I'M WORKING HARD.."
3025 T = 0
3030  FOR J = 1 TO X(2)
3040 A(J,2) = (1 + X(3)) ^ J
3050 A(J,3) = A(J,1) / A(J,2)
3060 T = T + A(J,3)
3070  NEXT J
3100  VTAB (8): CALL  - 958
3110  FOR J = 1 TO X(2):XD =  ABS (A(J,3)): GOSUB
300:A$(J) = X$
3115  IF A(J,3) < 0 THEN X$ = X$ + "-":A$(J) = X$
3120  PRINT "CASH FLOW - PERIOD ";J;" ==>$";X$
3130  NEXT J
3140  PRINT
3145 XD = T / X(1):XD =  INT (XD * 1000 + 0.5) / 1000
3150  PRINT "PROFITABILITY INDEX  ==> ";XD
3170  PRINT  TAB( 26)"------"
3200  VTAB (20): CALL  - 958: INPUT "WANT PRINTED
REPORT?(Y/N) ==> ";N$
3205  IF  LEFT$ (N$,1) = "Y" THEN 3220
3210  IF  LEFT$ (N$,1) = "N" THEN 3500
3215  GOSUB 60: GOTO 3200
3220  VTAB (4): CALL  - 958
3225  PRINT "DO THE FOLLOWING:"
3230  PRINT  TAB( 5)"1- LOAD PRINTER PAPER"
3235  PRINT  TAB( 5)"2- TURN ON PRINTER"
3240  PRINT  TAB( 5)"3- SET TOP-OF FORM"
3245  PRINT  TAB( 5)"4- PRINTER ON-LINE"
3250  PRINT : INPUT "TYPE RETURN KEY TO
CONTINUE....";N$
3255  VTAB (12): CALL  - 958: INPUT "PRINTER SLOT(1-7)
";SL
```

```
3260  IF SL > 0 AND SL < 8 THEN 3270
3265  GOSUB 60: GOTO 3255
3270  VTAB (14): CALL  - 958: PRINT "PRINTER TYPE:"
3275  PRINT  TAB( 5)"1- SERIAL"
3280  PRINT  TAB( 5)"2- PARALLEL"
3285  PRINT : GOSUB 55:SP = N
3290  IF N > 0 AND N < 3 THEN 3300
3295  GOSUB 60: GOTO 3270
3300  PR# SL
3305  IF SP = 2 THEN  PRINT  CHR$ (9);"80N"
3310  PRINT  TAB( 27)"PROFITABILITY INDEX REPORT"
3315  PRINT " "
3320  PRINT  TAB( 10)"1- INITIAL PROJECT COST
==>$";X(1)
3325  PRINT  TAB( 10)"2- # OF CASH FLOW PERIODS==>
";X(2)
3330  PRINT  TAB( 10)"3- INTEREST RATE (%)      ==> ";
INT (X(3) * 10000) / 100
3335  PRINT " "
3340  FOR J = 1 TO X(2)
3345  PRINT  TAB( 10)"CASH FLOW - PERIOD ";J;"
== $";A$(J)
3350  NEXT J
3355  PRINT " "
3360  PRINT  TAB( 10)"PROFITABILITY INDEX  ==> ";XD
3365  PRINT  TAB( 35)"------"
3370  PRINT  CHR$ (12)
3380  IF SP = 2 THEN  PRINT  CHR$ (9);"40N"
3390  PR# 0
3500  VTAB (20): CALL  - 958: INPUT "WANT ANOTHER
CALCULATION?(Y/N) ";N$
3510  IF  LEFT$ (N$,1) = "Y" THEN 1000
9000  PRINT : PRINT "PROGRAM CALLED = MAIN MENU"
9005 P$ = "ID.000"
9010  PRINT  CHR$ (4);"RUN ";P$;",D1"
```

圖12-12　獲利率程式

12.5　確定程式的正確性

在第二章的 2.10.2 節中我們曾經討論過了一個程式不同程度的正確性

；我們現在就把這些原則應用到我們的投資決策的程式中來。

第一層次的工作就是一個程式行一個程式行的找出它們語法中的錯誤並加以改正；在做完了這件事之後，我們就可以說在程式中已經沒有任何語法的錯誤了。

在做完了第一層次的改正工作之後，我們就選擇一組的測試資料輸入程式來確定第二及第三層次的正確性。測試資料的選擇最主要的是要使程式中的每一個敍述都能夠最少被執行一次，然後我們就使用手算的方式計算結果並且與程式中產生的結果相比較。這樣子我們就可以知道程式中也沒有了語意的錯誤，而這個程式也能夠正確地得到我們所要的結果。

在達成了第三層次正確性的要求之後，我們再選擇五組的有效資料重複地測試這個程式；這些資料是用來測試程式的不同層面的；在這個工作成功地完成之後，我們就可以說這個程式對於所有合法的資料都能正確工作了。

在 5.6 節中我們討論過了 Alpha/Beta 的測試觀點；在這個程式系統中，我們同時採用了這兩種方法。有兩名使用者證實了這個程式適合於Beta觀點的要求，他們在試用了 30 天之後都提出了正確的報告，而我們由他們的報告中得到這些程式都能夠適合於 Beta的要求。

至於第六層次正確性的要求則必須等待這個程式系統上市之後再做觀察了。

12.6 程式系統的記錄手册

這一節將前面的商業程式系統（投資決策）的記錄呈現給您。

接下去的幾節中將討論所有的作業方法及程序。

12.6.0 介紹的部份

簡　介：

這本手册有兩個目的，第一個目的就是教您如何使用投資決策（

"Investment Decisions")這個軟體系統，我們將使用實際的例子做詳盡的說明。第二個目的則希望這本手册成爲使用者的參考手册。

這個投資決策的程式系統是存在磁片上的，手册中共包含了十章，每一章討論系統中的一個程式。

您在使用這個程式系統之前，您必須具備著 32K 的 Apple II Plus，並且與一個磁碟機相連處於 DOS 3.3的狀況之下。

這本參考手册假設您對於使用整個硬體系統已經非常瞭解了；首先boot您的投資決策的磁片，您將在螢光幕上看到主手册（MAIN MENU），這個手册就如同本節後面顯示的一般。

對於資產的投資您必須先具有利率方面的概念，這樣子才能使您實際的使用這個軟體系統，我們在附錄B中會對這些觀念做一個大致地描述。在這個程式系統中的許多程式都與利率的計算有關，任何的貸款或是存款的帳目都必須定義4個基本的變數值。它們是；

1. 現值或貸款的數量。
2. 年利率。
3. 年數及月數。
4. 終值或定期付款值。

有了其中的三個值，而我們再定義一個值就可以計算出另外的那一個值了；這第5個值就是：

5. 複利計算的時間或是付款的時間。

計算年金終值的程式將依照您輸入的終值，每期的付款額、年利率、或是付款的次數等4個值中的三個值計算出另外一個值。

同時這個程式也計算出如果在加入了附加保險費後的年金終值；在這種情況下您必須輸入加入附加保險費後的費率。這時候您就可以看到新的終值及新的定額定期付款數了。

對於固定資產折舊的計算方法有四種公式可用，每一種我們都在程式中

列出供您選擇。第一種方法就是直綫法，其他的方法都是加速的法則。雖然我們知道對加速法而言有許多種不同的修正版正，但是在這裏使用的只是標準的型式。請參考附錄B。

至於加速法的目的是希望能夠減低稅金方面的開銷，一個公司在採用了加速法折舊的方式之後，它能夠儘快地將她的固定資產折舊掉。

我們這個投資決策的程式事實上就是在做財務分析的工作，它幫助一個公司決定如何才能使庫存的價值最大；在目前這麼許多的方法當中，我們提供了4種有效的計算投資的方法供您參考。

第一個就是償還點計算法（ Payback Method ）；雖然這種方法並不實際，但是因爲一般人都廣泛的使用它，所以我們也把它包含在內。至於其他的三種公式都是比較具有實際效用的，我們隨後就將介紹給您。未來回收額（Future returns ）的定義就是依照目前的投資額扣掉回收額的稅金，並且在折舊完畢之前所計算出來的結果；因此，回收額事實上就是從事這項投資之後的流入現金。

1. 償還點計算法（ Payback Method ）：得回當初投資額所需要的時間數。
2. 淨現值法（ Net Present Value(NPV) ）：未來回收額的現值扣掉投入資本的成本再減去當初的投資額所得的數目。
3. 投資報酬率（ Internal Rate of Return(IRR) ）：這是依據回收額現值與當初投資額的比例計算出來的利率。
4. 獲利率指標（ Profitability Index(PI) ）：利潤／本金得到的比例。回收額的現值除以原先投資額的現值計算而得。

```
                SYSTEMS HOUSE

             INVESTMENT DECISIONS

MAIN MENU:
          1 -LOAN SCHEDULES
          2 -SAVINGS SCHEDULES
REQUEST:  3 -ANNUITY SCHEDULES
          4 -DEPRECIATION SCHEDULES
          5 -AMORTIZATION SCHEDULES
          6 -APR SCHEDULES(eg AUTOS)
          7 -PAYBACK METHOD
          8 -NPV=NET PRESENT VALUE
          9 -IRR=INTERNAL RATE OF RETURN
          10-PROFITABILITY INDEX

          0 -EXIT SYSTEM

WHICH ONE? ==> 3

Is That Correct?(Y/N) = Y
```

這個時候您就被要求輸入一個選擇，您可以選擇手册中任何的數字（包括了零在內）。在您打入了您選擇的數字之後，按下 return 鍵執行這個被選擇的程式。如果您選擇的是零，那麼程式系統就結束執行了。至於輸入的數字超出了 0 到 10 的範圍，您就會在螢光幕底行的中間看到錯誤訊息的顯現，而您又回到選擇的螢光幕了。

12.6.1 第一章

第一章

貸款分期定額償還表 Loan Schedules - ID.0010

這個程式由 4 個副程式所組成，它們分別是：

```
MENU:
----
*** 1- REGULAR PAYMENT ON NOTE
    2- PRESENT VALUE ON NOTE
    3- # OF PAYMENTS
    4- ANNUAL INTEREST RATE

    0- EXIT PROGRAM

WHICH ONE? ==> 1

IS THAT CORRECT?(Y/N) = Y
```

在您選擇了執行這個程式之後，您將再被要求輸入以上四種選擇中的一個（也可以選擇零回到主程式去）；輸入您選擇的號碼並按下 return 鍵，接著您就會被詢問您是否確定要執行您的選擇。

對於這個程式所使用的數學公式可以參考附錄B。

1. 每期應還金額 (REGULAR PAYMENT ON NOTE)

這個程式將計算出在貸出一筆款之後，每一個單位時間內所應該攤還的固定數目。螢光幕的顯示情形將如下面一般。

```
        *** REGULAR NOTE PAYMENT ***


ANNUAL INTEREST RATE = 13.52
NOTE AMOUNT BORROWED = 6800.00
# OF YEARS,MONTHS    = 4,0
# OF COMPOUNDING PERIODS
    1- YEARLY              2- SEMIANNUALLY
    3- QUARTERLY           4- MONTHLY
    5- DAILY

WHICH ONE? ==> 4
```

這時螢光幕將被清除，然後再將您輸入的資料及計算得到的固定攤還數目顯示出來，並且在螢幕的底行顯示出"WANT NOTE SCHEDULE DISPLAYED? (Y/N) " 這樣的訊息。

您可以輸Y或是N做爲答覆。一個 "Y" 的答覆會使得帳目表依照年的次序而顯現出來； "N" 的答覆則不做這個顯示。

接著您將看到"WANT ANOTHER CALCULATION?(Y/N) "的訊息，一個 "Y" 的輸入會使程式從頭再開始執行；而 "N" 的輸入則使您回到主手册中。

您現在應該 boot 這個程式並且試試看上面的例子。

2. 帳上的現值 (PRESENT VALUE ON NOTE)

這個副程式將計算出帳上的現值，您必須輸入固定的付款額，付款的次數，以及年利率。

```
          *** PRESENT VALUE OF NOTE ***

 REGULAR PAYMENT ON NOTE =$184.19
 ANNUAL INTEREST RATE    = 13.52

 # OF YEARS,MONTHS       = 4,0
 # OF COMPOUNDING PERIODS
     1- YEARLY           2- SEMIANNUALLY
     3- QUARTERLY        4- MONTHLY
     5- DAILY

 WHICH ONE? ==> 4
```

螢光幕將被清除乾淨然後顯示出您輸入的資料及計算的結果。

```
     PRESENT VALUE OF NOTE    =$6800.00
```

現在您將看到"WANT NOTE SCHEDULE DISPLAYED?(Y/N) "的訊息。輸入一個 "Y" 字，您就會看到帳目表按照年的次序顯現出來。輸入 "N" 字則取消這個動作。

至於其他的部份就如前節討論的一般。

您應該試試看這個程式，感覺一下程式執行的情形。

3. 定期付款的次數 (# OF PAYMENTS)

這個副程式用來計算償還一筆貸款所需的付款次數。它是依據您輸入的固定付款數額、貸款的數額及年利率而計算的。

```
          *** # OF PAYMENTS ***

REGULAR PAYMENT        =$184.19
ANNUAL INTEREST RATE =  13.52
NOTE AMOUNT BORROWED =$6800.00

# OF COMPOUNDING PERIODS
    1- YEARLY              2- SEMIANNUALLY
    3- QUARTERLY           4- MONTHLY
    5- DAILY

WHICH ONE? ==> 4
```

這個程式將依照前面所描述的過程執行。

```
          # OF PAYMENTS          = 48
```

您現在最好利用上面例子中的資料試著執行這個程式。

4. 年利率 (ANNUAL INTEREST RATE)

這個副程式依照固定付款數額、貸款數額及付款次數計算出年利率。

```
        *** ANNUAL INTEREST RATE ***

REGULAR PAYMENT      =$184.19

NOTE AMOUNT BORROWED =$6800.00
# OF YEARS,MONTHS    = 4,0
# OF COMPOUNDING PERIODS
    1- YEARLY              2- SEMIANNUALLY
    3- QUARTERLY           4- MONTHLY
    5- DAILY

WHICH ONE? ==> 4
```

這個程式將依照前面所描述的方式執行。

```
ANNUAL INTEREST RATE = 13.52%
```

計算年利率的過程大約要花費幾秒鐘的時間，解決這個問題的最快速方法是使用數值分析中的 Power Series 逼近法；我們得到的答案將逼近至一個百分點的千分之一才結束計算的過程。

現在您就應該利用上面的資料試著執行這個程式了。

12.6.2 第二章

第二章

複利終值表－ Savings Schedules - ID.0020

這個程式也是由 4 個副程式所組成的，它們分別是：

```
MENU:
----
     1- FUTURE VALUE ON SAVINGS
     2- PRESENT VALUE ON SAVINGS
     3- # OF PAYMENTS
*** 4- ANNUAL INTEREST RATE

     0- EXIT PROGRAM

WHICH ONE? ==> 4
```

在您進入這個程式之後，您就被要求選擇上面的 4 個副程式中的一個（包括零－返回主程式－在內）。選擇了您所要執行的副程式之後，程式就會詢問您是否確定這個選擇，在您確定了之後，這個副程式才被執行。

至於程式中使用的數學方面的公式，請參考附錄B。

1. 期末本利總額 (FUTURE VALUE ON SAVINGS)

這個副程式將計算一個儲蓄帳戶的期末本利總額，依據您輸入的存款現值，存款年限，年利率及複利的計算方式計算而得。

```
            *** FUTURE VALUE OF SAVINGS ***


    PRESENT VALUE SAVINGS= 2500
    # OF YEARS,MONTHS     = 5,0
    ANNUAL INTEREST RATE = 5.25
    # OF COMPOUNDING PERIODS
        1- YEARLY               2- SEMIANNUALLY
        3- QUARTERLY            4- MONTHLY
        5- DAILY

    WHICH ONE? ==> 3
```

這時候螢光幕將被清除，並且顯現出您輸入的資料及儲蓄帳戶的期末本利總額。

```
          FUTURE VALUE SAVINGS = 3244.49
```

現在您將被詢問 "WANT ANOTHER CALCULATION?(Y/N) " 這樣的問題；輸入一個‘"Y"’字將使這個程式再從頭被執行；一個 "N"’字則使您回到主手册。

試著輸入這個例子所使用的資料到這個副程式內。

2. 存款現值 (PRESENT VALUE ON SAVINGS)

這個副程式將依據您輸入的未來預估值、年利率、儲蓄的年數及一年中計算複利的次數，計算出目前存款的現值。

```
*** PRESENT VALUE OF SAVINGS ***

FUTURE VALUE SAVINGS = 3244.49

# OF YEARS,MONTHS     = 5,0
ANNUAL INTEREST RATE = 5.25
# OF COMPOUNDING PERIODS
    1- YEARLY                2- SEMIANNUALLY
    3- QUARTERLY             4- MONTHLY
    5- DAILY

WHICH ONE? ==> 3
```

螢光幕將被清除乾淨，而您所輸入的資料及計算所得的儲蓄現值將顯示出來。

```
PRESENT VALUE SAVINGS= 2500
```

這時您將被詢問 "WANT ANOTHER CALCULATION?(Y/N) " ；您輸入一個 "Y" 字將使這個副程式重新再被執行；輸入一個‘"N"’字則使執行返回到主手册去。

請試著輸入資料執行這個副程式

3. 存款的時間數 （# OF PAYMENTS）

這個副程式將計算出在多少次複利計算週期後，您的存款可以達到期末本利總和值，您必須輸入存款的現值，未來預估值，年利率及一年中計算複利的次數。

```
*** # OF PAYMENT PERIODS ***

FUTURE VALUE SAVINGS = 3244.49
PRESENT VALUE SAVINGS= 2500

ANNUAL INTEREST RATE = 5.25
# OF COMPOUNDING PERIODS
    1- YEARLY                2- SEMIANNUALLY
    3- QUARTERLY             4- MONTHLY
    5- DAILY

WHICH ONE? ==> 3
```

螢光幕將被清除乾淨，並且將您輸入的資料及計算所得的次數顯示出來。

```
# OF PAYMENT PERIODS = 20
```

現在您將看到 "WANT ANOTHER CALCULATION?(Y/N) " 的訊息；輸入一個 "Y" 使程式從頭再被執行；輸入 "N" 字則返回到主手册。

請您輸入上面的資料試著執行這個副程式。

4. 年利率 （ANNUAL INTEREST RATE）

這個副程式將依據存款帳戶內的現值，未來價值，存款的年數，以及計算複利的方式，計算出年利率。

```
            *** ANNUAL INTEREST RATE ***

   FUTURE VALUE SAVINGS = 3244.49
   PRESENT VALUE SAVINGS= 2500
   # OF YEARS,MONTHS    = 5,0

   # OF COMPOUNDING PERIODS
       1- YEARLY              2- SEMIANNUALLY
       3- QUARTERLY           4- MONTHLY
       5- DAILY

   WHICH ONE? ==> 3
```

計算利率的工作是非常困難的，我們這裏使用的是 Power Series 逼近的方法求得我們所要的結果，它的精確度將達到一個百分點的千分之一。

這時螢光幕將被清除並且顯示出您輸入的資料及計算所得的年利率。

```
ANNUAL INTEREST RATE = 5.25 %
```

您將被詢問 "WANT ANOTHER CALCULATION?(Y/N) " 的問題；輸入一個 "Y" 就使程式從頭再開始執行；輸入一個 "N" 則使您回到主手册去。

請試試執行這個程式。

12.6.3 第三章

第三章

年金終值表 Annuity Schedules - ID.0030

這個程式由 4 個副程式所組成，它們分別是：

```
MENU:
----
    1- FUTURE VALUE OF ANNUITY
*** 2- PAYMENT ON ANNUITY
    3- # OF PAYMENTS
    4- ANNUAL INTEREST RATE

    0- EXIT PROGRAM

WHICH ONE? ==> 2
```

在您進入這個程式之後您就被要求選擇您要執行的副程式了（包括零－回到主程式－在內）；打入您選擇程式的號碼，再按下 return鍵，這時您就必須確定您的選擇，然後程式才會被執行。

數學方面的公式您可以在附錄B 中見到。

至於範例的報表則可以參考附錄C。

1. 年金終值（FUTURE VALUE OF ANNUITY）

這個程式依據您輸入的定期存款數額、付款的年數及月數、年利率、以及每年付款的次數計算出一個帳戶的年金終值。

```
           *** FUTURE VALUE OF ANNUITY ***

PAYMENT EACH PERIOD    = 178.73
# OF YEARS,MONTHS      = 24,0
ANNUAL INTEREST RATE = 7.75
PAYMENTS EACH YEAR     =
    1- YEARLY                2- SEMIANNUALLY
    3- QUARTERLY             4- MONTHLY
    5- DAILY

WHICH ONE? ==> 4
```

螢光幕將被清除乾淨並把您輸入的資料及計算得到的年金終值顯示出來。

```
FUTURE VALUE ANNUITY =$150000.81
                      -----------

FRONT LOADED?(Y/N)     = Y
PERCENT LOADING (%)    = 8.75

NEW PAYMENT/PERIOD     =$194.73
                        -----------
    ***** OR *****
NEW FUTURE VALUE       =$136875.74
                        -----------
```

現在您將看到 "WANT PRINTED REPORT?(Y/N) " 的訊息，打入 "N" 字將使您跳過下面的幾個步驟；輸入 "Y" 字，螢光幕將被清除，並且顯示出下面的訊息；

```
DO THE FOLLOWING:
    1- LOAD PRINTER PAPER
    2- TURN ON PRINTER
    3- SET TOP-OF-FORM
    4- PRINTER ON-LINE

TYPE RETURN KEY TO CONTINUE....

PRINTER SLOT(1-7) = 1

PRINTER TYPE:
    1- SERIAL
    2- PARALLEL

WHICH ONE? = 1
```

這時候您將被詢問 "WANT ANOTHER CALCULATION?(Y/N) " ，做 "Y" 的反應，使得這個程式再從頭開始執行；輸入 "N" 則使您回到主手册去。

現在請您輸入上面例子中的資料試試執行這個程式的結果。

2. 每期固定付款數額 (PAYMENT EACH PERIOD)

這個副程式將計算出每期固定的付款數額，依據的資料是期望的終值，年利率、年數及月數，以及每年的付款次數。

```
          *** PAYMENTS EACH PERIOD ***

FUTURE VALUE ANNUITY =$150000
# OF YEARS,MONTHS     = 24,0
ANNUAL INTEREST RATE = 7.75
PAYMENTS EACH YEAR    =
    1- YEARLY                2- SEMIANNUALLY
    3- QUARTERLY             4- MONTHLY
    5- DAILY

WHICH ONE? ==> 4
```

螢光幕將被清除並且將輸入的資料及計算的結果顯示出來。

```
PAYMENT EACH PERIOD   =$178.73
                        -----------

FRONT LOADED?(Y/N)    = Y
PERCENT LOADING (%)   = 8.75

NEW PAYMENT/PERIOD    =$194.73
                        -----------
    ***** OR *****
NEW FUTURE VALUE      =$136875.74
                        -----------
```

接著您將看到 "WANT PRINTED REPORT?(Y/N) " 的訊息，如果您回應一個 "N" 字，那麼您將跳過下面的幾個畫面；如果您打入 "Y" 字，螢光幕將被清除並且要求您做下列的工作；

```
DO THE FOLLOWING:
    1- LOAD PRINTER PAPER
    2- TURN ON PRINTER
    3- SET TOP-OF-FORM
    4- PRINTER ON-LINE

TYPE RETURN KEY TO CONTINUE....

PRINTER SLOT(1-7) = 1

PRINTER TYPE:
    1- SERIAL
    2- PARALLEL

WHICH ONE? = 1
```

現在您將被詢問 "WANT ANOTHER CALCULATION?(Y/N) " 的問題；打入 "Y" 使您再重新的執行這個程式；打入 "N" 則使程式回到主手册去了。

試著輸入上面例子中的資料執行這個程式。

3. 付款的次數 (# OF PAYMENTS)

這個副程式將依據每期固定的付款數額、年金終值、年利率、以及每年付款的次數計算出總付款的次數。

```
          *** # OF PAYMENT PERIODS ***

FUTURE VALUE ANNUITY =$150000
PAYMENT EACH PERIOD  =$178.73

ANNUAL INTEREST RATE = 7.75
PAYMENTS EACH YEAR   =
    1- YEARLY              2- SEMIANNUALLY
    3- QUARTERLY           4- MONTHLY
    5- DAILY

WHICH ONE? ==> 4
```

螢光幕將被清除並且顯示出輸入的資料以及應付款項的次數。

```
# OF PAYMENT PERIODS = 288
```

現在您就會看到 "WANT ANOTHER CALCULATION?(Y/N) " 的訊息；一個 "Y" 的回應使您重新再執行這個程式； "N" 的回應則使您看到主手册。

試著使用上面的資料執行這個程式。

4. 年利率 (ANNUAL INTEREST RATE)

這個副程式計算出年利率，依據的是每期固定的付款數額，年金終值，這個帳戶的持續年數，以及付款的方式。

```
          *** ANNUAL INTEREST RATE ***

    FUTURE VALUE ANNUITY =$150000
    PAYMENT EACH PERIOD  =$178.73
    # OF YEARS,MONTHS    = 24,0

    PAYMENTS EACH YEAR   =
        1- YEARLY              2- SEMIANNUALLY
        3- QUARTERLY           4- MONTHLY
        5- DAILY

    WHICH ONE? ==> 4
```

利率的計算較爲困難；我們這裏使用的是數值分析中的 Bisection 方法；整個計算的過程要等到精確度達到一個百分點的千分之一之後才結束。

這時候螢光幕將被清除並且顯示出輸入的資料及計算得到的年利率。

```
          ANNUAL INTEREST RATE = 7.75 %
```

現在您就會看到 "WANT ANOTHER CALCULATION?(Y/N) " 的訊息；打入 "Y" 字使您重新再執行這個程式；打入 "N" 字則使您回到主手册的部份去。

試著執行這個程式以便對它多一些瞭解。

12.6.4 第四章

第四章

折舊表（Depreciation Schedules ）- ID.0040

這個程式由下面四段所組成；它們分別是：

* 一直綫法（ Straight Line (SL)）
* 一倍數餘額遞減法（Double Declining Balance (DDB)）
* 一年數合計法（ Sum-of-Digits (SOD)）
* 一生產數量法（Units-of-Production (UOP)）

在您選擇了這個程式之後，您就被要求輸入下面的訊息：

```
PURCHASE PRICE =$25000
SALVAGE VALUE  =$3000
USEFUL YEARS   = 5
USEFUL HOURS   = 11000
```

接著程式就會被執行，而您就會在電腦使用上面的4種方法計算資料時在螢光幕上看到"．．．．．PROCESSING．．．．．"的訊息。這個程式所能處理的最大年數是30，至於錢數的極限則是$999,999.99。

電腦將花費一些時間來處理您輸入的資料，尤其是在數目較大或是年數較多的時候。當處理完畢之後，您將在螢光幕上看到產生的結果，以及 "WANT PRINTED REPORT(Y/N) " 的訊息。

打入一個 "Y" 字並按下return鍵，將使螢光幕再度被清除並且顯示出要求您準備好列表機的訊息。當設定列表機的工作完成了之後，您就被詢問 PRINTER SLOT(1-7) 的問題；接著您必須告訴電腦您的列表機是使用APPLE的串列介面卡或是平行輸出介面卡與 Apple 做聯繫的。在回答了以上的幾個問題之後，列表機就會把報表印出來了。

接著您又會看到 "WANT ANOTHER CALCULATION?(Y/N) " 的訊息

了；打入一個 "Y" 字使得程式重新被執行；打入 "N" 字則使您回到主程式。

請參看在附錄C中輸出報表的範例。至於數學方面的計算公式則在附錄B中可以看到。

12.6.5 第五章

第五章

攤銷表（Amortization Schedules - ID.0050）－

這個程式將印出一個攤銷表，它需要由使用者輸入一些資料。

這個程式將不會計算像是每月應付款或是其他的輸入資料；因爲計算分期定額償還表的程式就是計算這些資料的；如果您在使用這個程式之前，需要求得這些資料，那麼您就必須先執行計算貸款分期定額償還表的程式，然後使用產生的結果做爲這個計算程式的輸入資料。

程式的輸入部份：

```
          *** AMORTIZATION SCHEDULE ***

     ENTER TODAY'S DATE(MMDDYY) = 012081

     REGULAR PAYMENT = 184.19
     TERM IN YRS,MOS = 4,0
     PRINCIPAL AMOUNT= 6800
     % INTEREST/YEAR = 13.52
     # PAYMENTS/YEAR = 12
     BUYER'S NAME    = JL CAMPBELL
     SELLER'S NAME   = FRIENDLY CHEVROLET
     STARTING YEAR-YY= 81
     STARTING MO. -MM= 02
     STARTING DAY -DD= 15
```

當您將所有的資料都輸入了之後，您就被詢問“PRINTER SLOT(1-7)”以及列表機型式（串列的或是平行輸出的）的問題了。

在您回答了這些問題之後，攤銷表就印在紙上了。這個報表將告訴您每年付的利息總和，以及償還的本金數額，並且報表上的最後一個項目就是您要付的最後一筆款的數額。

電腦將花費一些時間處理這些資料使得它們能夠被印成報表；尤其當您輸入的數字很大，而年數又很多的時候，那就需要您的耐心了。

接著您將被詢問 "WANT ANOTHER CALCULATION?(Y/N)" 的問題如果您的回應是 "Y" ，那麼程式就從頭再被執行一次；如果回應是 "N"，您就回到主手册去了。

試著執行這個程式，瞭解它作業的情形。

至於範例的報表您可以在附錄C 中看到，而數學的公式則請參考附錄B。

12.6.6 第六章

第六章

APR 表（ APR Schedules ）- ID.0060

這個程式將印出 APR 表，它需要使用者輸入一些必須的資料。

這個程式將由輸入的資料中計算出每個月的付款數額。同時它也要求您輸入保險金的數額(Credit Life and Accident & Health insurance amounts)，接著程式就會將您輸入的資料及計算所得的眞實年利率顯示出來。然後您就可以選擇您所要計算的付款方式了。

程式中輸入：

```
          *** APR SCHEDULE ***

     ENTER TODAY'S DATE(MMDDYY) = 012081

     REGULAR PAYMENT = 184.19
     TERM IN YRS,MOS = 4,0
     PRINCIPAL AMOUNT= 6800
     ADD-ON INTEREST = 7.5
     PAYMENTS/YEAR   = 12
     CREDIT LIFE INS.= 211.91
     ACCIDENT/HEALTH = 373
```

在這個時候，螢光幕將顯示出輸入的資料及每月固定的付款額（按照不同的選擇）；它們分別是：

```
1- NOTE
2- NOTE + C.L.
3- NOTE + C.L. + A & H
```

計算眞實年利率的工作需要花費一些時間，它依據的是美國目前最風行的租借法計算得來的。

接著您將看到 "WANT SCHEDULE DISPLAYED?(Y/N) " 的問題，輸入‘"Y"’字使得帳目表一次顯示出12個月的資料。

至於回應一個 "N" 字則使您看到"WANT PRINTED REPORT?(Y/N)"的訊息。打入 "N" 字將使您跳過下面幾個步驟，打入 "Y" 字，您將被要求再做下面的輸入工作。它們是；

```
BUYER'S NAME     = JL CAMPBELL
SELLER'S NAME    = FRIENDLY TV
STARTING YEAR-YY= 81
STARTING MO. -MM= 02
STARTING DAY -DD= 15
```

接著您就必須選擇付款的方式了。

在所有的資料輸入完畢了之後，您還需要對“ PRINTER SLOT(1-7) ”的問題做回應，接著就是設定列表機型式的問題了（串列的或是平行輸出的）。

等到您對所有的問題都給了正確的答案之後，報表就在列表機上印出來了。報表中將列出每年付出的利息總和及償還的本金。跳頁的工作大約每隔三年做一次，等到所有的資料印完了之後，在結束的地方會有一個最後一次應付款的數額。

電腦在處理資料以便將它們印在紙上的工作中，必須花費一些時間，當資料的數字很大而年數很多的時候，就需要您耐心的等待了。

接著您又會被詢問 "WANT ANOTHER CALCULATION?(Y/N) ? 的問題，如果您輸入 "Y" 那麼程式就再度地被執行， "N" 則使您回到主手册。

請試著使用上面例子中的資料執行這個程式。至於範例的報表您可以在

附錄C中看到，也可以在附錄B中找到使用的數學公式。

12.6.7 第七章

第七章

償還點計算法（ Payback Method ）- ID.0070

這個程式將依照原先的投資數額。及每一期的流入現金數計算出在多少時間之後能夠收回原先的投資金額。這個程式所能允許的現金流入的期數爲 10。

在程式開始被執行的時候，您將被要求輸入兩個值；它們分別是；

```
        1- INITIAL PROJECT COST ==>$1000
        2- CASH FLOW PERIODS    ==> 6

 ANY CHANGES?(Y/N) N
```

螢光幕將顯示出下面的訊息；

```
         CASH FLOW FOR PERIOD 1 ==>$100
         CASH FLOW FOR PERIOD 2 ==>$200
         CASH FLOW FOR PERIOD 3 ==>$300
         CASH FLOW FOR PERIOD 4 ==>$400
         CASH FLOW FOR PERIOD 5 ==>$500
         CASH FLOW FOR PERIOD 6 ==>$600

 ANY CHANGES?(Y/N) N
```

在您將資料輸入完畢之後，電腦將顯示出 ".....PROCESSING....." 的訊息，告訴您它正在處理輸入的資料。

這個程式將花費一些時間做計算的工作，尤其在數字很大而期數又多的時候，更需要您的耐心了。等到計算完畢了之後，您就會在螢光幕上看到結

```
PAYBACK PERIOD ==> 4
```

接著您就會被詢問 "WANT PRINTED REPORT?(Y/N) " 的問題，如果您打入 "N" 字，那麼以下的幾個步驟就不會出現；如果您的回應是 "Y"，螢光幕將被清除，而您會看到下面的要求：

```
DO THE FOLLOWING:
    1- LOAD PRINTER PAPER
    2- TURN ON PRINTER
    3- SET TOP-OF-FORM
    4- PRINTER ON-LINE

TYPE RETURN KEY TO CONTINUE....

PRINTER SLOT(1-7) = 1

PRINTER TYPE:
    1- SERIAL
    2- PARALLEL

WHICH ONE? = 2
```

這個程式能夠處理不論是串列的或是平行的介面卡，如果您使用了其他的介面卡，那麼請您參考附錄A。

接著您再被詢問 "WANT ANOTHER CALCULATION?(Y/N)" 的問題；打入 "Y" 使程式重新再執行一次，打入 "N" 則使您又回到主手册。

試著使用上面例子中的資料執行這個程式。

至於範例的報表請參看附錄C，數學的公式則請參看附錄B。

12.6.8 第八章

第八章

淨現值法（ NPV=NET PRESENT VALUE ）- ID.0080

這個程式將計算出一項投資的淨現值，依據的資料是投資初值，流入現金的期數、以及利率。流入現金期數的最大值是10。

下面是三個您必須回答的問題，它們是；

```
1- INITIAL PROJECT COST ==>$1000
2- CASH FLOW PERIODS     ==> 6
3- INTEREST RATE (%)     ==> 10

ANY CHANGES?(Y/N) N
```

這時螢光幕將顯示出下面的敍述；

```
CASH FLOW FOR PERIOD 1 ==>$100
CASH FLOW FOR PERIOD 2 ==>$200
CASH FLOW FOR PERIOD 3 ==>$300
CASH FLOW FOR PERIOD 4 ==>$400
CASH FLOW FOR PERIOD 5 ==>$500
CASH FLOW FOR PERIOD 6 ==>$600

ANY CHANGES?(Y/N) N
```

在您將資料輸入完畢之後，您就會看到".....PROCESSING....." 的訊息，告訴您現在正在做計算資料的工作。

電腦將花費一些時間在計算的工作上，尤其當數字較大而期數又多的時候，更是需要您的耐心了。在計算工作完畢了之後，螢光幕上就會顯示出計算的結果：

```
TOTAL ALL CASH FLOWS ==>$      1,403.94
INITIAL PROJECT COST ==>$      1,000.00
                         -------------
NET PRESENT VALUE    ==>$        403.94
```

接著您將被詢問 "WANT PRINTED REPORT?(Y/N) " 的問題，如果您的回應是 "N" ，那麼您將跳過下面的幾個步驟；輸入一個 "Y" 字，則您就會看到下面的訊息；

```
DO THE FOLLOWING:
    1- LOAD PRINTER PAPER
    2- TURN ON PRINTER
    3- SET TOP-OF-FORM
    4- PRINTER ON-LINE

TYPE RETURN KEY TO CONTINUE....

PRINTER SLOT(1-7) = 1

PRINTER TYPE:
    1- SERIAL
    2- PARALLEL

WHICH ONE? = 2
```

這個程式可以處理 Apple][的串列或是平行的介面卡，如果您使用了其他的介面卡，那麼請您參考附錄A。

接著您又會看到 "WANT ANOTHER CALCULATION?(Y/N)" 的訊息，如果您的回應是 "Y" ，程式將再被執行一次，如果是 "N" ，那麼您就回到主手册內了。

請試著執行這個程式，以使您更加地瞭解它。

範例的報表請參看附錄C，至於數學的公式則在附錄B中有詳細地說明。

12.6.9 第九章

第九章

投資報酬率（ IRR=INTERNAL RATE OF RETURN ）- ID.0090

這個程式將計算出投資報酬率（ IRR ），依據的資料是初期投資額以及流入現金的期數。並且規定流入現金的期數最多不能超過10期。

下面是兩個首要的問題；

```
        1- INITIAL PROJECT COST ==>$1000
        2- CASH FLOW PERIODS    ==> 6

ANY CHANGES?(Y/N) N
```

這時螢光幕將顯示出下面的幾個敍述：

```
CASH FLOW FOR PERIOD 1 ==>$100
CASH FLOW FOR PERIOD 2 ==>$200
CASH FLOW FOR PERIOD 3 ==>$300
CASH FLOW FOR PERIOD 4 ==>$400
CASH FLOW FOR PERIOD 5 ==>$500
CASH FLOW FOR PERIOD 6 ==>$600

ANY CHANGES?(Y/N) N
```

在您將資料輸入完畢之後，電腦將顯示出".....PROCESSING....."的訊息，告訴您現在正在處理您輸入的資料。

處理資料的工作必須花費一些時間，尤其是數字比較大而期數又多的時候，您更需要耐心地等待了。在計算完畢之後，螢光幕上會顯示出來計算的結果。

```
INTERNAL RATE OF RETURN ==> 19.53%
```

接著您就被詢問"WANT PRINTED REPORT?(Y/N) "的問題；如果您的回應是 "N" ，那麼下面的幾個步驟就被跳過，如果回應是 "Y" ，您就被要求做下面的工作；

```
DO THE FOLLOWING:
    1- LOAD PRINTER PAPER
    2- TURN ON PRINTER
    3- SET TOP-OF-FORM
    4- PRINTER ON-LINE

TYPE RETURN KEY TO CONTINUE....

PRINTER SLOT(1-7) = 1

PRINTER TYPE:
    1- SERIAL
    2- PARALLEL

WHICH ONE? = 2
```

這個程式只能夠處理 Apple][的串列或是平行的介面卡，如果您使

用了其他的介面卡，那麼請您參考附錄A中的說明。

在您印完了報表之後，您就會看到"WANT ANOTHER CALCULATION? (Y/N)"的訊息；打入 "Y" 字使得程式重新再被執行，打入 "N" 字則使您回到主手册。

請利用上面例子中的資料執行這個程式，相信對您會有很大的幫助的。

範例的報表請參看附錄C；附錄B中有您所要的數學公式。

12.6.10 第十章

第十章

獲利率指標（PROFITABILITY INDEX）- ID.0100

這個程式將計算出一項投資的獲利率指標，依據的資料是初期投資額、流入現金的期數、以及利率。程式中規定流入現金的期數不可超過10期。

首先您必須輸入下面的三項資料；

```
1- INITIAL PROJECT COST ==>$1000
2- CASH FLOW PERIODS    ==> 6
3- INTEREST RATE (%)    ==> 10

ANY CHANGES?(Y/N) N
```

這時螢光幕將顯示出下面的幾個敍述；

```
CASH FLOW FOR PERIOD 1 ==>$100
CASH FLOW FOR PERIOD 2 ==>$200
CASH FLOW FOR PERIOD 3 ==>$300
CASH FLOW FOR PERIOD 4 ==>$400
CASH FLOW FOR PERIOD 5 ==>$500
CASH FLOW FOR PERIOD 6 ==>$600

ANY CHANGES?(Y/N) N
```

在您輸入了全部的資料之後，您就會看到".....PROCESSING....."

的訊息，告訴您輸入的資料正在被處理。

電腦在處理資料的過程中將花費一些時間，尤其當數目較大而期數又多的時候，您必須耐心地等待。等到計算的工作完成了之後，螢光幕將顯示出計算的結果。

```
PROFITABILITY INDEX ==> 1.404
```

接著您必須對 "WANT PRINTED REPORT?(Y/N) " 的問題做回應；如果打入 "N" 字，將使您跳過下面的步驟；如果打入 "Y"，字，您將看到下面的要求；

```
DO THE FOLLOWING:
    1- LOAD PRINTER PAPER
    2- TURN ON PRINTER
    3- SET TOP-OF-FORM
    4- PRINTER ON-LINE

TYPE RETURN KEY TO CONTINUE....

PRINTER SLOT(1-7) = 1

PRINTER TYPE:
    1- SERIAL
    2- PARALLEL

WHICH ONE? = 2
```

這個程式能夠處理 Apple][的平行或是串列介面卡，如果您使用了其他的介面卡，那麼請參考附錄A。

接著的問題 "WANT ANOTHER CALCULATION?(Y/N)"，如果您回應 "Y"，程式將再被執行一次；如果是 "N"，則使您回到主手册。

請利用上面例子的資料執行這個程式。

範例報表在附錄C中可以看到；至於數學的公式則必須參考附錄B。

12.6.11 附　錄

附 錄 A

列表機的問題

問題 1：

任何為 Apple 所設計的軟體系統並不能完全地適用於所有的介面卡；我們的這項軟體是為了一般的介面裝置所設計的，例如 Apple 公司的 Silentype 列表機，具有 RS232C 的列表機，或是具有 APPLE II 串列及平行介面卡的列表機。如果您有其他廠商設計的介面卡，像是SSM或CCS，我們就無法保證印表的工作會正確地進行了。

問題 2：

如果您的 RS232 介面卡尚未將傳送速度（ baud rate ），跳頁的格式等設定好，那麼報表可能無法正確地印出；因此請您在印報表之前先檢查這些設定的值是否正確。

問題 3：

我們將所有報表的格式都安排成一行 80 個字，如果您使用的是較寬的紙，那麼在報表的右邊將留下空白的位置。

問題 4：

當您在APPLE Ⅲ上執行這個 APPLE Ⅱ版本的程式時，您必須將 APPLE Ⅱ的串列或是平行介面卡插在 APPLE Ⅲ的一個擴充接點上（最好

是第1個）；當您執行這個程式的時候，回應的就是 APPLE Ⅲ的擴充接點數目。至於列表機介面卡的型式則依照實際的情形回答。

問題5：

如果您執行這個程式的 APPLE Ⅲ版本，那麼您必須在串列介面卡與 RS-232介面卡（ Port C ）之間使用 MODEM 做爲傳輸的工具。使用 Silentype 列表機時，您必須將列表機連接在 Port A上。

問題6：

有一些列表機，像是Centronics 779-x及737-x 並不能接受跳頁（ chr$(12) ）的命令；因此，在印攤銷表或是 APR 表時可能無法印得很整潔；至於其他的報表則需要您控制跳行的動作，以便達到跳頁的效果。

附 錄 B

理論及數學公式

理 論：

所有的程式都是依照由上往下的結構方式設計的，並且也依照同樣的方式連接在一起。

所有的計算工作都是依照被大家所接受的公式而達成的。而這些公式都經過了許多次的檢查及實際應用後才能夠具有肯定的價值，而這也是我們採用它們的原因。

這個程式系統的操作步驟已經是盡量地簡單了，我們希望它能夠使您在使用時覺得非常地方便。

我們在計算應得款或是應付款時所考慮的利率都是複利形式的。例如下面計算一個存款帳戶的表就是一個例子；這個例子中存款額爲$1000.00 ，

年利率爲5％，而每季結算一次，那麼下表就是頭一年中四個季的帳目表。

#	本金	利息	總數
1	$1,000.00	$12.50	$1,012.50
2	1,012.50	12.66	1,025.16
3	1,025.16	12.81	1,037.97
4	1,037.97	12.97	1,050.94

由這個表中您可以看出來這個帳戶的存款數額每一季都依照利率累積計算，而下一季的數額則依照前一季的累積金額計算而得；這也就是複利的計算方式。至於這個程式系統中使用的計算公式就在隨後的部份介紹給您。

如果您想要詳細地瞭解所有的公式，那麼您必須參考財務管理，以及簿記方面的書籍。

數學公式

貸款分期定額償還表（ Loan Schedules ）：

MP ＝一定本金定期定額償還每期應償金額（ Regular Payment ）
i ＝年利率（ Interest Rate, Annual ）
c ＝計算複利的單位時間（ Compounding Periods ）
Y ＝年數（ Years ）
M ＝月數（ Months ）
PV ＝本金（ Principal Amount ）

```
  Y = (y * 12 + M)/12

           PV * (1/c)
 MP = ----------------------
      (1 - (1 + i/c)^(-Y*c)
```

定期存款到期本利和表（ Savings Schedules ）：

FV ＝複利終值（ Future value ）

PV ＝現值（Present Value）
i ＝複利（Interest）
c ＝計算複利的單位時間（Compounding Periods）
Y ＝年數（Years）
M ＝月數（Months）

```
 Y = (Y * 12 + M)/12
FV = PV * (1 + i/c)^(Y*c)
```

年金終值表（Annuity Schedules）：

PV ＝定期付款額（Payment Value）
FV ＝複利本利和（Future Value）
i ＝複利（Interest）
c ＝每年付款的次數（Payments/year）
Y ＝年數（Years）

```
FV = PV * (1 + i/c) * ((1 + i/c)^(Y*c) - 1)/(i/c)
```

折　舊（Depreciation）：

YD ＝年度折舊（Yearly Depreciation）
PP ＝買價（Purchase Price）
SV ＝殘值（Salvage Value）
YR ＝耐用年限（Useful Years）
HR ＝耐用時數（Useful Hours）

注意：若 SV <= 10%，則該項物品已無殘值存在。（遵照美國 IRS 的標準）。

直綫法 (Straight Line) :

```
        (PP - SV)
YD =    -------
          YR
```

倍數餘額遞減法 (Double Declining Balance) :

```
        (PP - SV)  2.0
YD =    ------- * ---
          YR       YR
```

年數合計法 (Sum-of-Digits) :

SUM = 由 1 到 YR 的年數合計值。
i = 某一特定的年數。

```
        (YR + 1)-i
YD =    ---------- * PP
           SUM
```

生產數量法 (Units-of-Production) :

```
        (PP - SV)
YD =    -------
           HR
```

攤銷表 (Amortization Schedules) :

注意：這個表是由貸款分期定額償還表的公式計算出來的。

APR 表 (APR Schedules) :

注意：這個表是由貸款分期定額償還表的公式計算得來的。

投資指數（Investment Indices）：

償還點計算法（Payback Method）：

這個方法計算出初期投資——所有流入現金總和＝0的那一點。

淨現值法（Net Present Value）：

NPV＝淨現值（Net Present Value）
Rn ＝第n次期間流入的現金（Cash Flow-period n）
C ＝投資現值（Initial Investment）
i ＝折扣率（利率）（Discount Rate(interest)）

$$NPV = \sum_{j=1}^{n} \frac{Rj}{(1+i)^j} - C$$

投資報酬率（Internal Rate of Return）：

$$IRR = \sum_{j=1}^{n} \frac{Rj}{(1+i)^j} = C$$

注意：i 精確至每一個百分點的千分之一； IRR 的值與投資現值的誤差在$10.00之內。

獲利率指標（Profitability Index）：

PI ＝獲利率指標（Profitability Index）

$$PI = \frac{\displaystyle\sum_{j=1}^{n} \frac{Rj}{(1+i)^{j}}}{C}$$

注意：PI＝1.0表示位於損益兩平點（ breakeven point ）。PI ＞1.0表示有獲利的可能。PI ＜1.0表示無利可圖。

附　錄　C

範例報表

```
                    ANNUITY REPORT

FUTURE VALUE OF ACC'T=$150000
PAYMENT EACH PERIOD  =$178.73
# OF PAYMENT PERIODS = 288
THE ANNUAL INTEREST  = 7.75 %
PAYMENTS EACH YEAR   = 12

PERCENT LOADING (%)  = 8.75

NEW PAYMENT/PERIOD   =$194.37
                       ---------

     *** OR ***

NEW FUTURE VALUE     =$136875.74
                       ---------
```

```
                DEPRECIATION REPORT

PURCHASE PRICE = $ 25000.00
SALVAGE VALUE  = $  3000.00
USEFUL YEARS   =  5
USEFUL HOURS   =  11000
```

==

YEAR	SL	DDB	SOD	UOP
1	4400.00	10000.00	8333.33	4400.00
2	4400.00	6000.00	6666.67	4400.00
3	4400.00	3600.00	5000.00	4400.00
4	4400.00	2160.00	3333.33	4400.00
5	4400.00	1296.00	1666.67	4400.00
	----------	----------	----------	----------
	22000.00	23056.00	25000.00	22000.00

MORTGAGE AMORTIZATION TABLE

DATE: 01/20/81

BUYER: JL CAMPBELL SELLER: FRIENDLY CHEVROLET

PRINCIPAL $6800 AT 13.52% FOR 4 YEARS 0 MONTHS.

REGULAR PAYMENT = $184.19

DATE	INTEREST	PRINCIPAL	BALANCE	TOTAL INTEREST
3/15/1981	$76.61	$107.58	$6,692.42	$76.61
4/15/1981	75.40	108.79	6,583.63	152.01
5/15/1981	74.18	110.01	6,473.62	226.19
6/15/1981	72.94	111.25	6,362.37	299.13
7/15/1981	71.68	112.51	6,249.86	370.81
8/15/1981	70.42	113.77	6,136.09	441.23
9/15/1981	69.13	115.06	6,021.03	510.36
10/15/1981	67.84	116.35	5,904.08	578.20
11/15/1981	66.53	117.66	5,787.02	644.73
12/15/1981	65.20	118.99	5,668.03	709.93
	-----------	-----------		
TOTAL 1981	$709.93	$1,131.97		
1/15/1982	$63.86	$120.33	$5,547.70	$773.79
2/15/1982	62.50	121.69	5,426.01	836.29
3/15/1982	61.13	123.06	5,302.95	897.42
4/15/1982	59.75	124.44	5,178.51	957.17
5/15/1982	58.34	125.85	5,052.66	1,015.51
6/15/1982	56.93	127.26	4,925.40	1,072.44
7/15/1982	55.49	128.70	4,796.70	1,127.93
8/15/1982	54.04	130.15	4,666.55	1,181.97
9/15/1982	52.58	131.61	4,534.94	1,234.55
10/15/1982	51.09	133.10	4,401.84	1,285.64
11/15/1982	49.59	134.60	4,267.24	1,335.23
12/15/1982	48.08	136.11	4,131.13	1,383.31
	-----------	-----------		
TOTAL 1982	$673.38	$1,536.90		
1/15/1983	$46.54	$137.65	$3,993.48	$1,429.85
2/15/1983	44.99	139.20	3,854.28	1,474.84
3/15/1983	43.42	140.77	3,713.51	1,518.26
4/15/1983	41.84	142.35	3,571.16	1,560.10
5/15/1983	40.24	143.95	3,427.21	1,600.34
6/15/1983	38.61	145.58	3,281.63	1,638.95
7/15/1983	36.97	147.22	3,134.41	1,675.92
8/15/1983	35.31	148.88	2,985.53	1,711.23
9/15/1983	33.64	150.55	2,834.98	1,744.87
10/15/1983	31.94	152.25	2,682.73	1,776.81
11/15/1983	30.23	153.96	2,528.77	1,807.04
12/15/1983	28.49	155.70	2,373.07	1,835.53
	-----------	-----------		
TOTAL 1983	$452.22	$1,758.06		

MORTGAGE AMORTIZATION TABLE

DATE: 01/20/81

BUYER: JL CAMPBELL SELLER: FRIENDLY CHEVROLET
PRINCIPAL $6800 AT 13.52% FOR 4 YEARS 0 MONTHS.
REGULAR PAYMENT = $184.19

DATE	INTEREST	PRINCIPAL	BALANCE	TOTAL INTEREST
1/15/1984	$26.74	$157.45	$2,215.62	$1,862.27
2/15/1984	24.96	159.23	2,056.39	1,887.23
3/15/1984	23.17	161.02	1,895.37	1,910.40
4/15/1984	21.35	162.84	1,732.53	1,931.75
5/15/1984	19.52	164.67	1,567.86	1,951.27
6/15/1984	17.66	166.53	1,401.33	1,968.93
7/15/1984	15.79	168.40	1,232.93	1,984.72
8/15/1984	13.89	170.30	1,062.63	1,998.61
9/15/1984	11.97	172.22	890.41	2,010.58
10/15/1984	10.03	174.16	716.25	2,020.61
11/15/1984	8.07	176.12	540.13	2,028.68
12/15/1984	6.09	178.10	362.03	2,034.77
	-----------	-----------		
TOTAL 1984	$199.24	$2,011.04		
1/15/1985	$4.08	$180.11	$181.92	$2,038.85
2/15/1985	2.05	181.92	.00	2,040.90
	-----------	-----------		
TOTAL 1985	$6.13	$362.03		
LAST PAYMENT =	$183.97			

** LOAN SCHEDULE REPORT **

DATE: 01/20/81

BUYER: JL3CAMPBELL SELLER: FRIENDLY TV

PRINCIPAL $7384.91 AT 7.5 % ADD-ON FOR 4 YEARS 0 MONTHS.

REGULAR PAYMENT = 200.01

DATE	INTEREST	PRINCIPAL	BALANCE	ACCUM INTEREST
3/15/1981	83.20	116.81	7,268.10	83.20
4/15/1981	81.88	118.12	7,149.98	165.09
5/15/1981	80.56	119.45	7,030.53	245.65
6/15/1981	79.21	120.80	6,909.73	324.86
7/15/1981	77.85	122.16	6,787.57	402.71
8/15/1981	76.47	123.54	6,664.03	479.18
9/15/1981	75.08	124.92	6,539.10	554.26
10/15/1981	73.66	126.34	6,412.76	627.93
11/15/1981	72.25	127.76	6,285.00	700.18
12/15/1981	70.81	129.20	6,155.80	770.99
TOTAL 1981	770.99	1,229.10		
1/15/1982	69.36	130.64	6,025.15	840.35
2/15/1982	67.88	132.13	5,893.02	908.23
3/15/1982	66.38	133.62	5,759.40	974.62
4/15/1982	64.88	135.12	5,624.28	1,039.51
5/15/1982	63.37	136.64	5,487.64	1,102.88
6/15/1982	61.83	138.18	5,349.46	1,164.71
7/15/1982	60.27	139.74	5,209.72	1,224.98
8/15/1982	58.70	141.31	5,068.41	1,283.67
9/15/1982	57.10	142.91	4,925.50	1,340.78
10/15/1982	55.49	144.52	4,780.98	1,396.26
11/15/1982	53.87	146.14	4,634.84	1,450.13
12/15/1982	52.22	147.79	4,487.04	1,502.35
TOTAL 1982	731.37	1,668.75		
1/15/1983	50.55	149.45	4,337.59	1,552.90
2/15/1983	48.87	151.14	4,186.44	1,601.77
3/15/1983	47.17	152.83	4,033.60	1,648.94
4/15/1983	45.45	154.56	3,879.04	1,694.39
5/15/1983	43.70	156.31	3,722.73	1,738.09
6/15/1983	41.94	158.07	3,564.66	1,780.03
7/15/1983	40.16	159.85	3,404.81	1,820.19
8/15/1983	38.36	161.64	3,243.16	1,858.55
9/15/1983	36.54	163.46	3,079.69	1,895.09
10/15/1983	34.70	165.31	2,914.39	1,929.79
11/15/1983	32.84	167.17	2,747.21	1,962.63
12/15/1983	30.95	169.06	2,578.16	1,993.58
TOTAL 1983	491.23	1,908.89		

** LOAN SCHEDULE REPORT **

DATE: 01/20/81

BUYER: JL CAMPBELL SELLER: FRIENDLY TV

PRINCIPAL $7384.91 AT 7.5 % ADD-ON FOR 4 YEARS 0 MONTHS.

REGULAR PAYMENT = 200.01

DATE	[illegible]ST	PRINCIPAL	BALANCE	ACCUM INTEREST
1/15/1984	[illegible]05	170.96	2,407.20	2,022.63
2/15/1984	2[illegible].12	172.89	2,234.31	2,049.75
3/15/1984	25.17	174.84	2,059.47	2,074.92
4/15/1984	23.20	176.81	1,882.66	2,098.12
5/15/1984	21.21	178.80	1,703.86	2,119.33
6/15/1984	19.20	180.81	1,523.05	2,138.53
7/15/1984	17.15	[illegible]35	1,340.20	2,155.69
8/15/1984	15.10	184.91	1,155.29	2,170.79
9/15/1984	13.02	186.99	968.30	2,183.82
10/15/1984	10.91	189.10	779.20	2,194.72
11/15/1984	8.78	191.23	587.97	2,203.50
12/15/1984	6.62	193.39	394.58	2,210.12
TOTAL 1984	216.54	2,183.58		
1/15/1985	4.44	195.56	199.02	2,214.57
2/15/1985	2.24	199.02	00	2,216.81
TOTAL 1985	6.69	394.58		

LAST PAYMENT = 201.26

```
                    PAYBACK METHOD REPORT

1- INITIAL PROJECT COST    ==>$1000
2- # OF CASH FLOW PERIODS==> 6

CASH FLOW - PERIOD 1 ==>$        100.00
CASH FLOW - PERIOD 2 ==>$        200.00
CASH FLOW - PERIOD 3 ==>$        300.00
CASH FLOW - PERIOD 4 ==>$        400.00
CASH FLOW - PERIOD 5 ==>$        500.00
CASH FLOW - PERIOD 6 ==>$        600.00

PAYBACK PERIOD EQUALS==> 4
```

```
                NPV - NET PRESENT VALUE REPORT

1- INITIAL PROJECT COST    ==>$1000
2- # OF CASH FLOW PERIODS==>$6
3- INTEREST RATE (%)       ==> 10

CASH FLOW - PERIOD 1 ==>$         90.90
CASH FLOW - PERIOD 2 ==>$        165.28
CASH FLOW - PERIOD 3 ==>$        225.39
CASH FLOW - PERIOD 4 ==>$        273.20
CASH FLOW - PERIOD 5 ==>$        310.46
CASH FLOW - PERIOD 6 ==>$        338.68

TOTAL FOR ALL PERIODS==>$      1,403.94
INITIAL PROJECT COST ==>$      1,000.00
                           -------------
NET PRESENT VALUE    ==>$        403.94
```

```
                  IRR - INTERNAL RATE OF RETURN REPORT

1- INITIAL PROJECT COST   ==>$1000
2- # OF CASH FLOW PERIODS==> 6

CASH FLOW - PERIOD 1 ==>$        100.00
CASH FLOW - PERIOD 2 ==>$        200.00
CASH FLOW - PERIOD 3 ==>$        300.00
CASH FLOW - PERIOD 4 ==>$        400.00
CASH FLOW - PERIOD 5 ==>$        500.00
CASH FLOW - PERIOD 6 ==>$        600.00

INTERNAL RATE EQUALS ==> 19.53%
```

```
                       PROFITABILITY INDEX REPORT

1- INITIAL PROJECT COST   ==>$1000
2- # OF CASH FLOW PERIODS==> 6
3- INTEREST RATE (%)      ==> 10

CASH FLOW - PERIOD 1 ==>$         90.90
CASH FLOW - PERIOD 2 ==>$        165.28
CASH FLOW - PERIOD 3 ==>$        225.39
CASH FLOW - PERIOD 4 ==>$        273.20
CASH FLOW - PERIOD 5 ==>$        310.46
CASH FLOW - PERIOD 6 ==>$        338.68

PROFITABILITY INDEX  ==> 1.404
                         ------
```

12.7 總　結

在第十二章中，我們向您介紹了一個商業上使用的程式系統，我們將這個系統內所有的程式列及測試等資料都在這一章中列出來供您研究了。

對這個程式系統做總體介紹的時候，我們把所有程式中應用的數學公式都列了出來，也對爲什麼使用這些公式及使用這些公式所產生的結果做了詳盡的說明。這使您在參考程式列的時候，能夠很快地瞭解它的意義。最後，我們將測試的步驟及程式的記錄都呈現在您的眼前。

我們將所有的程式列都包含在 12.4 節中；至於程式的記錄則在 12.6 節可以看到，這兩節也就包含了這個程式系統所有完整的資料了。

附錄A

APPLESOFT Ⅱ 命令

ABS(-3.14159)

求得括弧內元素的絕對值。上面的例子得到 3.14159 。

ASC("QUEST")

得到括弧內元素內第一個字的 ASCII 碼。上面的例子得到 81 (ASCII 的Q)。

ATN(2)

求得括弧內元素的反正切函數(arctangent)值(弳度量)。上面的例子得到 1.10714872 。

CALL 位置

CALL-922•

使得位於記憶體內由“位置”處開始存放的機器語言副程式被執行。上面的例子在螢光幕上產生跳行的動作。

CHR$(65)

求得括弧內元素所對應的 ASCII 字，元素的值必須在 0 與 255 之間。上面的例子求得 A 字。

CLEAR

將所有的變數清爲零，將所有的字串清除掉，並把先入後出暫存器（ stack ）清除乾淨。

COLOR=12

設定繪低解析度圖形的顏色。在上面的例子中，顏色被設定爲綠色。當您使用 GR 命令時，顏色自動被設定爲 0。下面是顏色與相對應數字的對照表。

0 black	4 dark green	8 brown	12 green
1 magenta	5 grey	9 orange	13 yellow
2 dark blue	6 medium blue	10 grey	14 aqua
3 purple	7 light blue	11 pink	15 white

想要找出螢光幕上某一點的顏色，您可以使用 SCRN 命令達到這個目的。

CONT

如果程式的執行被 STOP, END, ctrl—C 或是 reset 0G return 命令所打斷，CONT 命令使程式繼續由下一個指令開始執行（如 GOSUB 命令一般）。但是在 reset 0G 命令之後，再使用 CONT 命令，程式可能無法正確地工作，這是因爲有些程式指標及先入後出暫存器已經被清除掉了的關係。CONT 命令在下面的情形不能使用。

(a) 您已經將程式修改過了，或是
(b) 在停止執行的時候得到了錯誤訊息。

COS(2)

得到元素的餘弦函數（COSIN）值，這個值是弳度量值。在上面這個例子中，我們得到－.415146836。

CTRL-C

可以用來停止一個程式的執行或是列表的動作。如果它是第一個輸入的字，那麼它也可以用來打斷 INPUT 指令的動作（但是必須按下RETURN鍵）。

CTRL-X

通知 APPLE][取消剛打入的一行，但是並不取消這一行前面的部份。當您使用這個命令時，被取消的這一行的後面會出現一個（/）號。

DATA JOHN SMITH,"CODE 32",23.45,-6

用來建立一系列的資料，供READ敍述使用。在這個例子中，第一個資料就是文字 JOHN SMITH；第二個資料則是字串 "CODE 32"；第三個資料就是實數 23.45；第四個就是整數－6。

DEF FN A(W)=2*W+W

這個命令允許使用者在程式中定義一行的函數。首先您必須用 DEF 定義函數，然後您在程式的其他地方就可以使用這個函數了。上面的例子就是定義一個函數FN A(W)，稍後在程式中您就可以使用像是FN A(23)或

FN(-7*Q+1)等等的式子使用它了。

FN A(23)會使得函數中的W值用 23 來代替，而這個函數就變成2 * 23 + 23 = 69 了。至於 FN(-7*Q+1) 的式子，如果我們假設Q = 2，那麼這個式子就等於 FN A(-7*2+1) 或是 FN(-13) 了，而這個函數就被計算為2 *（－13）+（－13）或是－26－13或是－39。

DEL 23,56

將指定範圍內的程式行自程式中清除掉。在上面這個例子中，行號 23 到 56 的程式行都將被清除掉；至於想要清除某一個程式行的方法就是打入 DEL 350,350 之類的命令，或者只是打入這個行號，然後按下 RETURN 鍵就可以了。

DIM AGE(20,3), NAME$(50)

當執行一個 DIM 命令的時候，它會使電腦預留由零到指定的值那麼多個位置，供這個矩陣使用。在上面的例子中， NAME$(50) 就可以容納 51 個任何長度的字串；至於 AGE(20,3) 則可以存 21 * 4 = 84 個實數。

如果您在程式中並沒有使用 DIM 命令定義一個矩陣，那麼它最多只能使用11個元素（設定值為 10 ）。當您執行 RUN 或是 CLEAR 命令的時候，矩陣內所有的元素值都被清為零。

DRAW 4 AT 50,100

將存在造型表內的第 4 個造型繪在由座標 x = 50，y = 100 開始的螢光幕上。在 DRAW 被執行之前， COLOR 、 ROT 及 SCALE 的值必須先被定義。

END

使一個程式的執行停止，並且將控制權交回給使用者，但並不印出任何訊息。

EXP(2)

求得e的次方值。由於e＝2.718289，所以上面的例子得到7.3890561。

FLASH

使得由電腦輸出的字在螢光幕上閃動。您可以使用 NORMAL 命令使顯示的情形返回到白字黑底的正常狀況。

FOR

```
FOR W=1 to 20: ... :NEXT W
FOR Q=2 to -3 STEP -2: ... :NEXT Q
FOR Z=5 to 4 STEP 3: ... :NEXT
```

允許您設計一個迴圈執行在 FOR 與 NEXT 之間的敍述若干次。在第一個例子中，迴圈內的敍述將被執行 20 次，直到 W=21 時爲止。第二個例子告訴您如何使用 STEP 命令；至於第三個例子您可以看出在迴圈內的敍述將只被執行一次。

FRE(0)

告訴您目前您還可以使用的記憶體數（以 byte爲單位）。您放在括弧內的值並不重要，只要它是合法的APPLESOFT][值就可以了。這個命令同

時將不再被使用的字串自記憶體內清除掉。

GET ANS$

由鍵盤上得到一個輸入的字，但是並不將它顯示在螢光幕上，也不需要您按 RETURN 鍵。在上面的例子中，打入的字存在 ANS$ 內。

GOSUB 250

使程式的執行轉移到指定的行號上（例子中就是 250）。但是當執行到 RETURN 指令時，程式就轉移回到 GOSUB 指令下面的一個敘述去執行了。

GOTO 250

使程式的執行轉移到指定的行號上（例子中就是 250）。

GR

設定低解析度圖形的顯示狀況（40 乘 40 的螢光幕），並且在螢幕的底端留下 4 行文字資料行。執行這個指令之後，螢光幕被清為黑色，游標移到文字資料幕(text window)，而 COLOR 值被設定為 0。

HCOLOR=4

設定高解析度圖形的顏色。下面是顏色與相對應數字的對照表：

0 black1	4 black2
1 green (depends on TV)	5 (depends on TV)
2 blue (depends on TV)	6 (depends on TV)
3 white1	7 white2

HGR

只能適用在具有硬式 APPLESOFT][（存在 ROM 內）的機器上；設定 280 × 160 的高解析度螢幕，並在下方留下 4 行的文字資料幕。當執行這個指令時，螢光幕被清爲黑色，而記憶體內的第一頁（ page 1 ）被顯示出來，但是 HCOLOR 値及存顯示文字資料的記憶體並不受到影響。

HGR2

設定全螢幕的高解析度圖形顯示狀況（ 280 乘 192 ）；當執行這個指令時，螢光幕被清爲黑色，而記憶體的第二頁被顯示出來。至於儲存顯示文字資料的記憶體並不受到影響。

HIMEM: 16384

設定可以被 APPLESOFT][程式使用的最高記憶體位置。它是被用來保護儲存著資料、高解析度圖形、或是機器語言程式的記憶體。這個命令並不會被 CLEAR, RUN, NEW, DEL 等等指令，以及 reset 改變的。

HLIN 10, 20 AT 30

在低解析度圖形狀況下繪出一條水平綫，使用的顏色是前面被指定的 COLOR 。對於低解析度圖形而言，原點（ x = 0 ， y = 0 ）就是在螢光幕左上角的那一點。在這個例子中，我們由 x = 10 到 x = 20 （在 y = 30 的座標上）繪出一條水平綫。

HOME

將游標移到文字資料幕的左上角位置，同時並將資料幕內所有的文字清

除掉。

```
HPLOT 10,20
HPLOT 30,40 TO 50,60
HPLOT to 70, 80
```

上面這些指令在高解析度圖形狀況下，使用先前定義的 HCOLOR 繪出點或綫。顯示幕的原點就是（ x = 0 ，y = 0 ）；第一個例子在 x = 10 ，y = 20 的位置上繪出一個點。第二個例子則由 x = 30 ，y = 40 這一點繪一條直綫到 x = 50，y = 60 的點爲止。至於第三個例子則由最後被繪出的一個點開始繪一條直綫到 x = 70，y = 80 的點爲止；使用的顏色就是先前被繪出的那個點的顏色，而不必是最後被定義的 HCOLOR 。

HTAB 23

將游標依照指定的值往左或右移動（ 1 到 40 ）。在這個例子中，游標將位於第 23 行的位置上。

```
IF    情況       THEN    處理
IF AGE<18 THEN A=0: B=1: C=2
IF ANS$="YES" THEN GOTO 100
IF N<MAX THEN 25
IF N<MAX GOTO 25
```

如果跟隨在 IF 後面的運算式是“眞”，那麼就執行 THEN 後面的指令；否則，程式就往下一個行號繼續執行了。字串都是依照它的字母順序計算比較的。上面例子中的第 3 及第 4 個例子都是相同的意義。

INPUT

```
INPUT A%
INPUT "TYPE AGE THEN A COMMA THEN NAME ";B,C$
```

在第一個例子中， INPUT 敍述印出一個問號，等待您輸入一個整數；在第二個例子中的 INPUT 敍述將引號內的訊息全部顯示出來，然後等待您輸入一個數字（放入變數B內），一個逗號，以及一個字串（存入字串變數 C$ 內）。

INT(NUM)

求得小於或等於括弧內元素的最大整數。在上面的例子中，如果 NUM = 2.389 ，那麼您將求得2這個值。如果 NUM = -45.123345 ，那麼得到的結果就是－46 。

INVERSE

設定螢光幕的顯示狀況爲黑字白底。使用 NORMAL 命令可以使顯示情形成爲白字黑底的正常狀況。

IN#4

指定做輸入工作的外圍設備與電腦連接的擴充接點數目（ 1到 7 ）。使用 IN# 0 命令就可以使輸入狀況返回到鍵盤了。

LEFT$(...)

LEFT$("APPLESOFT][",5)

執行這個指令可以得到一個字串由左邊算起到指定數目爲止的所有的字。在上面的例子中，您將得到 APPLE 。

LEN("AN APPLE A DAY")

求得括弧內字串的長度（最多不可超過 255），例子中您得到 14。

LET

```
LET A=23.567
A$="DELICIOUS"
```

將“＝”號右邊的字串或是運算式的值放入“＝”號左邊的變數內。LET 是可以省略的。

LIST

```
LIST
LIST 200-3000
LIST 200,3000
```

第一個例子使得所有的程式行都被顯示在螢光幕上。第二個例子則使行號 200 到 3000 的程式行被顯示出來。如果您想要將由程式起頭到行號200的程式行列出，那麼您可以使用 LIST -200 命令達到目的；至於想要將由200到程式結束的程式行列印出來，使用的命令則是 LIST 200- 。第三個例子與第二個例子做同樣的工作。您可以使用 CTRL C 結束列印的動作。

LOAD

由磁帶將 APPLESOFT][的程式讀入記憶體內；您必須在執行 LOAD 命令之前先將磁帶轉到起頭的地方，並且按下磁帶機上的 play 鍵。當 LOAD 的動作完成了之後，電腦將會再發出“嗶”的一聲通知您程式

已經被存到記憶體內了。只有使用 reset 才能夠打斷這個動作。

LOG(2)

求得括弧內元素的自然對數值。例子中您將得到 .693147181 這個值。

LOMEM: 2060

設定一個 BASIC 程式所能使用最低記憶體的位置，這個指令可以保護變數不被高解析度圖形所損害。

MID$(...)

```
MID$("AN APPLE A DAY",4)
MID$("AN APPLE A DAY",4,9)
```

這個指令用來求得一個字串的部份字串。在第一個例子中，您可以得到括弧內字串的部份字串（由第 4 個字開始到結束為止）： APPLE A DAY 。在第二個例子中，則由第 4 個字開始取出 9 個字： APPLE A D 。

NEW

清除目前存在記憶體內的程式及所有的變數。

NEXT

請參看 FOR...TO...STEP 的部份。

NORMAL

使螢幕顯示返回到正常的情況下。

NOTRACE

結束 TRACE 的動作。請參看 TRACE 命令。

ON 運算式 GOSUB 行號1、行號2、……

ON ID GOSUB 100, 200, 4005, 500

依照運算式的值決定 GOSUB 轉移的行號。在上面的例子中，如果 ID=1，則執行 GOSUB 100 ；如果 ID=2 ，則執行 GOSUB 200 ；其餘的行號也是依照 **ID** 的值而決定。如果運算式的值是零或是超過行號的數目，那麼電腦就會執行下一個敍述。

ON ID GOTO 100, 200, 23, 4005, 500

與 ON ID GOSUB 命令相同，但是這個指令是依照 **ID** 的值而執行一個 GOTO 命令就是了。

ONERR GOTO 500

這個指令用來避免使程式停止執行的錯誤；當程式在執行中發生錯誤時，這個 ONERR GOTO 命令會使得程式的執行跳到指定的行號去(500)。

PDL(3)

得到指定遊戲控制器目前的值（在0與255之間）；遊戲控制器的號碼可以由0到3。

PEEK(37)

求得指定數元組（十進位的位置）內的資料（十進位）。上面這個例子就是求得位置37的數元組內的資料。

PLOT 10, 20

在低解析度狀況下，繪出一個在指定位置上的點。在這個例子中就是在 x = 10，y = 20的座標上繪出一個點，使用的 COLOR 就是先前定義的顏色。如果沒有定義則是黑色（COLOR = 0）。

POKE -16302,0

將一個十進位值（在第二個元素中指出）的對應二進位值放入第一個元素所指定的數元組內（在這個例子中就是－16302）。

POP

這個命令使存在先入後出暫存器內的一個 RETURN 位置被移出先入後出暫存器。因此在 POP 被執行後下一個被執行的 RETURN 命令會使程式的執行轉移到第2個 GOSUB 命令的下一個敍述去。

POS(0)

您可以使用這個指令得到游標目前所在的水平位置。而您放在括弧內的值並沒有任何關係，只要它能夠被 APPLESOFT][接受就可以了。

```
PRINT
PRINT A$; "X = ";X
```

第一個例子造成一個跳行的動作。如果您想要在螢光幕的固定位置上印出資料，那麼這些資料之間必須用逗號分開；如果您想將不同的資料連接在一起，那麼您就必須使用分號了。像第二個例子中，假設 A$ 內存的是 CORE ；而X內存的是3，執行這個敍述的結果就是在螢光幕上印出 COREX = 3 這樣的訊息。

PR#2

將輸出的訊息轉移到指定的擴充接點上（ 1到 7 ）。使用 PR#0 就可以使輸出轉移到螢光幕上了。

READ

```
READ A, B%, C$
```

執行這個指令時，電腦就從 DATA 內把資料讀入 READ 指令的元素內。在上面這個例子中， DATA 敍述的前兩個資料必須是數字，而第三個必須是字串，它們將分別的放入 A, B% 及 C$ 內。

RECALL MX

這個命令由磁帶內將一個整數的或是實數的矩陣讀入程式內。一個矩陣可以使用不同的名字 RECALL 它，但是您必須先在程式內使用 DIM 命令定義它就是了。當您執行 RECALL 或是 STORE 命令的時候，您不必指出矩陣的標註（ Subscripts ）；在上面這個例子中，存在磁帶內的矩陣元素將被讀入 MX(0) 、 MX(1) 、……內，至於多餘的矩陣元素則不受到影響。您在執行 RECALL 之前，必須先按下磁帶機的 play 鍵；當開始及結束這個讀入矩陣的動作的時候，電腦都會發出“嗶”的一聲。只有 reset 能夠使 RECALL 命令被打斷。

REM

REM THIS IS A REMARK

這個命令允許您在程式中加入文字資料做爲註記。

RESTORE

將指到 DATA 敍述內資料的指標設定爲第一個元素，這使得下一個 READ 命令由 DATA 敍述的第一個資料開始讀起。

RESUME

在處理錯誤的程式段（參看 ONERR GOTO ）結束的時候，使程式繼續由發生錯誤的敍述開始執行。

RETURN

這個指令使程式由最近被執行的 GOSUB 指令的下一個敍述繼續執行。

RIGHT$(....)

RIGHT$("SCRAPPLE",5)

由括弧內字串的最右邊一個字開始取到指定數目的字爲止。在上面這個例子中， APPLE 被取得。

RND

RND(1)

取得一個大於等於 0 或小於 1 的實亂數（ random real number ）。使用 RND(0) 可以取得剛剛被產生出來的亂數值。當您在括弧內使用負的元素時，只要每一次元素值相同，那麼產生的亂數就是相等的，並且使隨後採用正值做爲元素的 RND 命令依照一定的程序產生出亂數來。只要您在使用 RND 命令之前沒有使用負的元素值，那麼每一次產生的數值都是介於 0 與 1 之間的新數目。

ROT = 16

設定由 DRAW 或 XDRAW 指令所繪出造型的旋轉角度。 ROT=0 使得造型繪在原先定義的位置上； ROT=16 使得造型往順時鐘的方向旋轉 90 度。至於整個的旋轉可以由 ROT=64 重新開始的。

RUN 500

將所有的變數、指標及先入後出暫存器清除掉，並由指定的行號開始執行程式（在這個例子中就是行號 500 ）。如果您不指出行號，那麼程式就由最低的行號開始執行。

SAVE

將一個程式存到磁帶內；您必須在執行 SAVE 命令之前按下 record 及 play 鍵， SAVE 命令並不會檢查您按的鍵是否正確，因此您必須正確的操作。當開始存及存完的時候，您都會聽到“嗶”的一聲。

SCALE=50

設定由 DRAW 或 XDRAW 命令所繪出的圖形的大小。 SCALE=1 使得圖形完全依照造型表而繪出圖形， SCALE =255 則使造型擴大255倍。注意： SCALE=0 是一個造型擴大的極限。

SCRN(10,20)

在低解析度圖形狀況下，您可以使用這個命令求得某一個點的顏色。在上面的例子中，您可以得到座標 x=10,y=20 這一個點的顏色。

SGN(NUM)

如果括弧內的元素是一個負值，那麼您可以得到－1 ；當元素值是零的時候，得到的就是 0 ；如果元素值是正值，那麼您就可以得到 1 。

SHLOAD

由磁帶將一個造型表存入記憶體內，您必須在執行這個命令之前，先設定好 HIMEM: 值，這樣子才能使造型表存在 HIMEM: 以下的記憶體內，而不至於被程式或變數破壞。

SIN(2)

求得一個角的正弦（ sine ）弳度量值。在這個例子中就是 .909297427 。

SPC(8)

必須被使用在 PRINT 敍述內；這個指令用來設定前一個被印出的項目與下一個要被印出項目中間所留的空格。但是您在使用它時前面必須加上一個分號。

SPEED = 50

設定字輸送到螢光幕或是其他輸入／輸出設備的速度。最低的速度是 0，最高的則是 255 。

SQR(2)

取得括弧內元素的平方根值；在這個例子中您可以求得 1.41421356 。SQR 執行的速度較 ^.5 命令要快得多。

STOP

使得一個程式停止執行，並且印出停止執行的行號。

STORE MX

將一個整數或實數的矩陣存到磁帶上去。使用者必須自己按下 record 及 play 鍵（在執行 STORE 命令之前）。在結束存入的動作的時候，電腦會發出“嗶”的一聲。在這個例子中， MX(0), MX(1), MX(2),.. 都會被存到磁帶內，但是對於 MX 這個變數而言並不受到任何的影響。請參看 RECALL 命令。

STR$(...)

STR$(12.45)

求得括弧內元素的字串表示。在這個例子中，您得到字串“ 12.45 ”。

TAB

TAB(23)

必須使用在 PRINT 敍述中，括弧內的元素值只能在 0 與 255 的範圍之內；如果元素值較目前游標所在的位置數要大，那麼 TAB 就會將游標移往指定的位置上去；計算的方式是由目前游標所在的顯示行的最左邊開始計算。

如果您指定的元素值較游標目前所在位置的數目要小，那麼游標就不會被移動。 TAB(0) 會把游標放在第 256 個位置上。

TAN(2)

求得元素的正切函數(tan)值(必定是弳度量)。在這個例子中，您得到－2.18503987。

TEXT

將螢幕設定爲顯示文字資料的狀況，同時也可以將文字資料幕擴增爲全螢幕的顯示情形。

TRACE

使得每一個正在被執行敍述的行號顯示在螢光幕上。 TRACE 命令不能夠被 RUN 、 CLEAR 、 NEW 、 DEL 或 reset 取消，您只能使用 NOTRACE 取消它的作用。

USR(3)

這個函數將其中的元素值傳送到機器語言的副程式內。元素值會被放到浮點的計算器內(位置 $9d 到 $A3)，並且執行一個 JSR 命令，將執行跳到 $0A 的位置內；位置 $0A 到 $0C 內必須存有一個 JMP 的命令，使這個機器語言的副程式能被執行。至於由副程式計算得到的值也是被放在浮點計算器內。

VAL("-3.7E4A5PLE")

將一個字串轉變成一個實數或是整數值。如果字串的第一個字不是數字，那麼您就會得到一個零。在上面的例子中，您得到的值就是－37000。

```
VLIN 10,20 AT 30
```

在低解析度情況下依照先前指定的 COLOR 值繪出一條垂直綫。這條直綫繪在第三個元素指定的行上；像上面的例子，我們可以由螢光幕上看到一條由 y = 10 到 y = 20 繪出的垂直綫（位在 x = 30 的行上）。

VTAB

```
VTAB(15)
```

這個指令用來垂直地移動游標在螢光幕上的位置。頂端一行的數目是 1，而底端行的數目則是 24 。

WAIT

```
WAIT 16000,255
WAIT 16000,255,0
```

允許一個有條件的停止被加在程式中。第一個元素是一個記憶體位置，我們測試這個數元組內的某些數元是 1 或是 0 。第二個元素的二進位值就是表示那一個數元是您要測試的；1 表示您要測試這個數元，0 則表示不必去理會它。至於第三個元素值不是 0 就是 1，它表示您希望測試的數元的情況：1 表示這個數元必須是 0，0 表示這個數元是 1 。如果您沒有指定第三個元素，那麼電腦假設它是 0 。

當由第二個元素指定的數元是依照第三個元素的意思表現的時候，那麼這個 WAIT 命令就停止執行了。

XDRAW

XDRAW 3 AT 180,120

將造型表內的第三個造型在x = 180，y = 120的座標上繪出；當您使用這個命令繪出造型時，繪出的每一個點的顏色都與原先存在的那個位置上的點的顏色互成補色。這使得我們能夠輕易地將 XDRAW 命令繪出的圖形，使用 XDRAW 消除掉，而不至於影響到底色。

附錄B

整數BASIC的命令

ABS（運算式） 範例：ABS(X)

求得括弧內元素及運算式的絕對值。

ASC（字串） 範例：ASC("BACK")
ASC(A$)

求得括弧內字串內第一個字的 ASCII 碼（十進位值）。

AUTO（數目） 範例：AUTO 10

設定自動增加行號數目的功能。由您指定的數目開始計算程式的行號，並以 10 做爲增加的間隔。您可以使用 CTRL-X ，再打入 MAN 命令按下 return 鍵取消 AUTO 狀況。

AUTO 數字1、數字2 範例：AUTO 10,5

與上面的作業方式相同，只是行號是以 5 做爲增加的單位的。

CALL 運算式　　　　　　　　範例：CALL-922

這個命令用來執行一個機器語言的副程式，這個副程式的開始位置（十進位）由您指定。上面這個例子會產生跳行的動作。

CALL 運算式　　　　　　　　範例：CALL-936

這個命令將螢光幕上的文字資料清除乾淨，並把游標移到文字資料幕的左上角位置。

CLR

將目前所有的 BASIC 變數及尚未設行列值的矩陣清除乾淨，但對程式本身並沒有任何影響。

COLOR 運算式　　　　　　　　範例：COLOR=12

設定低解析度圖形的顏色。在上面的例子中，顏色被設定爲綠色。當執行 GR 指令時，圖形的顏色自動地被設定爲黑色。下面是顏色與相對應的數字：

0 black	4 dark green	8 brown	12 green
1 magenta	5 grey	9 orange	13 yellow
2 dark blue	6 medium blue	10 grey	14 aqua
3 purple	7 light blue	11 pink	15 white

您也可以使用 SCRN 命令找出螢光幕上任何一點的顏色。

CON

將一個由 CTRL-C 打斷的程式繼續執行，程式中所有的變數值並不會被改變。

CTRL

CTRL-C

我們可以使用這個命令打斷一個程式執行或是列印的動作，也可以使用它打斷 INPUT 敍述的執行（**CTRL-C** 必須是輸入的第一個字）。

CTRL-X

通知 Apple][取消剛被輸入記憶體內的一個程式行，其他的程式部份並不受到影響。在被取消的這一行後面您可以看到一個 (/) 符號。

DEL 數目 1 範例：DEL 27

DEL 數目 1、數目 2 範例：DEL 23,56

前面的例子取消數目 1 所代表的程式行；後面的例子取消由數目 1 開始到數目 2 結束的所有程式行。

DIM 變數（運算式） 範例：DIM AGE(20)
DIM NAME$(50)

DIM 命令使得 Apple][爲您指定的變數預留記憶體的位置。當一個 DIM 敍述被執行的時候，電腦自動地會預留由零到您指定數目爲止的記憶體

。在上面的例子中， NAME$(50) 將有 50 個位置可以使用，這是由於一個字串所能使用的範圍是在 1 到 255 之間的原因。至於 AGE(20) 這個矩陣則會有 21 個佔用兩個數元組的元素可以使用了。

DSP 變數　　　　範例：DSP IX:DSP B$

這是一個檢錯時使用的命令，它能將您指定的變數在每次改變的時候使它同時與引起這個改變的程式行的行號顯示出來。（注意： RUN 命令將取消 DSP 這個命令的功用，因此您在使用它的時候，必須使用 CON 或 GOTO 命令執行程式。）

END

使一個程式停止執行，並把控制權還給使用者。程式停止的時候並不會顯示任何訊息，只會送出一個整數 BASIC 的系統標示 ">" 到螢光幕上。

FOR 變數＝運算式 1 TO 運算式 2 STEP 運算式 3

範例：
```
FOR W=1 to 20:...:NEXT W
FOR Q=2 to -3 STEP -2:...:NEXT Q
FOR Z=5 to 4 STEP -1:...:NEXT
```

這個結構允許您設計一個可以被反覆執行許多次的迴圈，所有在 FOR 與 NEXT 之間的敍述將依照您指定的次數執行。第一個例子使用 W 來計算執行迴圈中敍述的次數，當 W = 1,2,3,...20 的時候，這些敍述都將被重覆地執行，直到迴圈結束時（ W=21 ）爲止。第二個例子則告訴您如何使用 STEP 計算迴圈次數的間隔。至於檢查執行迴圈是否完成的工作是在迴圈的結尾處做的，因此在第三個例子中隱藏在迴圈內的敍述將被重覆執行 4 次。

GOSUB 運算式　　　　範例：
```
GOSUB 250
GOSUB 500+10*X
```

使程式的執行轉移到您指定行號的敍述去(例子中就是250行號的敍述)。當一個 RETURN 指令被執行的時候，程式再被轉移回到剛被執行的 GOSUB 敍述的下一個敍述去執行。

GOTO　運算式　　　　範例：GOTO 25*10

使程式的執行立刻轉移到指定行號的敍述去。上面的例子中我們使程式轉移到250行號上去執行。

GR

設定混合式的圖形狀況；將螢幕清爲黑色，並且預留40乘40個顯示點的位置(可以使用的顏色有15種)，同時在螢光幕的底端預留4行的文字資料幕。

HIMEM　運算式　　　　範例：HIMEM:24576

設定 BASIC 程式所能使用的最高記憶體位置。

HLIN　運算式1，運算式2 AT 運算式3
　　範例：HLIN 10,20 AT 30

使用前面指定的 COLOR 值在低解析度圖形情況下繪出一條水平綫。對於顯示幕而言，原點(x = 0，y = 0)就是在螢幕的左上角。在上面的例子中，這一條水平綫就是由 x = 10 繪到 x = 20 的位置(位於 y = 30 的列上)；對於這一條水平綫的另外一個說法就是我們由(10，30)這一點繪一條直綫到(20，30)這一點爲止。

IF 運算式 THEN 敍述

```
IF AGE < 18 THEN A=0: B=1: C=2
IF ANS$ = "YES" THEN GOTO 100
IF N < MAX THEN 25
IF N < MAX THEN GOTO 25
```

如果跟隨在 IF 後面的運算式是“眞”（不是零），那麼跟隨在 THEN 後面同一行中的指令都將被執行，否則程式將跳到下一個行號去執行。範例 3 及 4 都是代表同樣的意義。

INPUT 變數1,變數2, 字串 範例：INPUT A
INPUT "TYPE AGE COMMA NAME ",B,C$

在第一個例子中，INPUT 敍述先印出一個問號，並且等待您由鍵盤輸入一個數字，這個數字將被放入變數 A 內。在第二個例子中，INPUT 敍述將包含在引號內的字串印出，並等待您輸入一個數字、一個逗號、以及一個字串，分別將數字放入 B 內，字串放入 C$ 內。

IN# 運算式 範例：IN# 4

指定輸入設備與電腦連接的介面卡位在的擴充接點數目。IN# 0 使得輸入由鍵盤發生。

LEN （字串） 範例：LEN(B$)

將 B$ 這個字串內所擁有的字數求得，字串長度的範圍是在 0 與 255之間的。

LET 範例： LET A=23
A$="DELICIOUS"

將“＝”號右邊的字串或是運算式的值放入“＝”號左邊的變數內。LET是可以被省略的。

LIST

LIST

LIST 數目1 範例：LIST 200

LIST 數目1，數目2 範例：LIST 200,3000

第一個例子使整個的程式列被顯示在螢光幕上；第二個例子則將行號200的程式行顯示出來；第三個例子把由數目1到數目2中間所有的程式行顯示出來。

LOAD

由磁帶將程式讀入記憶體內。使用者必須在執行這個命令之前先將磁帶轉回頭，並按下 play 鍵。當電腦在磁帶上找到這個訊息的時候，會發出“嗶”的一聲，而在讀入記憶體的動作完成了之後也會發出“嗶”的一聲。您只有使用 reset 打斷 LOAD 的動作。

LOMEM 運算式 範例：LOMEM:4096

與 HIMEM: 指令相同，只是它是用來設定 BASIC 所能使用的最低記憶體位置就是了。

MAN

取消 AUTO 命令的功用。

MOD

取得一個除式的餘數。

NEW

將目前存在記憶體內的 BASIC 程式及所有的變數清除掉。

NEXT

請參看 FOR...TO...STEP 部份的討論。

NO DSP 變數 範例：NO DSP X

取消指定變數的 DSP 功能。

NO TRACE

取消 TRACE 情況。

TAB 23

在螢幕上往左或往右移動游標到指定的行上（ 1 到 40 ）。在這個例子中，游標將被移到 23 行的位置。

PDL （運算式） 範例：PDL(3)

取得指定遊戲控制器目前的值（0到255之間）。遊戲控制器共有0到3四個編號。

PEEK （位置） 範例：PEEK(37)

取得記憶體內指定位置（十進位）內所存資料的十進位值。對於超過32767 位置的記憶體，您必須使用相對應的負值表示。

PLOT 運算式1，運算式2 範例：PLOT 10, 20

在低解析度狀況下，在指定的位置上繪出一個點。上面的例子就是在x = 10，y = 20的座標上繪出一個點。至於這個點的顏色決定在前面所指定的 COLOR 值。

POKE 位置，值 範例：POKE -16302,0

將第二個元素的值存入指定的記憶位置內。

POKE 50,127

設定螢幕顯示的狀況爲黑字白底。

POKE 50,255

設定螢幕顯示的狀況爲白字黑底。

POP

將存在先入後出暫存器內的 RETURN 位置移開，這使得其後遇到的 RETURN命令將執行轉移到第二個 GOSUB 命令的下一個敍述去。

PRINT 範例：PRINT
PRINT A$;"X = ";X

第一個例子造成一個跳行的動作；如果在一個 PRINT 敍述中有許多不同的項目，那麼這些項目必須用逗號分開（分別在不同的停留位置上印出）；如果您想將這些項目接連著印出來，那麼您就必須使用分號做爲分界的符號了。上面的例子中，如果 A$ 內存的是 "CORE" ，而 X＝3 ，則您將在螢光幕上看到 COREX = 3 的訊息。

PR# 擴充接點數目 範例：PR#2

與 IN# 相似，只是它將輸出轉移到由 1 到 7 中的一個擴充接點上就是了。 PR#0 使輸出的動作重返螢光幕上。

REM 範例：REM THIS IS A REMARK

允許您將文字資料加入到程式中做爲一個註記。

RETURN

使得程式的執行轉移到最近被執行的 GOSUB 指令的下一個敍述去。RETURN 同時結束一個副程式的執行。

RND （運算式） 範例：RND(5)

得到一個整數的亂數值，這個亂數值大於等於 0 而小於 5 。如果運算式是一個負值，那麼您得到的亂數就是介於 0 與運算式值加 1 之間的一個值。

RUN

將所有的變數清除爲零，並且由最低的行號開始執行程式。

RUN 運算式 範例：RUN 500

將所有的變數、指標及先入後出暫存器清除，並且由您指定的行號開始執行程式（例子中就是 500 ）。

SAVE

將一個程式存到磁帶上去；使用者必須在執行這個命令之前按下 record及 play 鍵， SAVE 命令並不檢查這個動作。當開始與結束記錄的動作時，您都會聽到一聲“嗶”的聲音。

SCRN （運算式 1 ，運算式 2 ） 範例：SCRN(10,20)

在低解析度圖形狀況下求得螢光幕上一個指定點的顏色。在上面的例子中，我們求得的就是 x = 10 ， y = 20 這一個點的顏色值。運算式 1 及運算式 2 的值必須在允許的範圍內。

SGN （數字） 範例：SGN(X)

求得一個數字的正負號；如果元素值是負值，那麼您得到的就是－1 ；

如果是零，得到的就是 0 ；若您得到 1 的時候，元素必定是正值。

TAB 運算式　　　　範例： TAB 21
TAB I+24

將游標移到您所指定的螢光幕水平位置上，運算式的值必須在 1 與 40 之間。

TEXT

將螢光幕設定爲顯示文字資料的標準狀況（共 24 行，每行 40 個字），並且將文字資料幕設定爲全螢幕的狀況。

TRACE

使得每一個正在被執行的敍述的行號顯示在螢光幕上。這個命令您只能使用 NO TRACE 結束它，而 RUN, CLEAR, NEW, DEL 或 reset 這些命令對它都不會發生作用。

VLIN 運算式 1 , 運算式 2 AT 運算式 3
範例： VLIN 10,20 AT 30

在低解析度情況下，依照前面設定的 COLOR 值在螢光幕上繪出一條垂直線。這一條線繪出的位置就是由這個命令的第三個元素值設定的。上面的例子中，這一條線就是在 x = 30 的行上由 y = 10 繪到 y = 20 的點上爲止。

VTAB 運算式　　　　範例：VTAB 15

將游標移到指定位置的顯示行上。顯示行的最高位置是 1 ，而最低位置是 24 。 VTAB 只能上下的移動游標，而無法左右的移動它。

附錄C

POKE, PEEK 及 CALL

POKE命令

POKE 32,L

依照L的值設定顯示幕的最左邊界限（0到39），0是最左邊的位置。

顯示幕的寬度並不因爲這個命令而改變，這就是說右邊的界限也會依照左邊界限的移動數量而移動。爲了要能夠保存您的程式及必要的資料，您在改變左邊界限值之前，最好先適當地設定螢幕的寬度值。

POKE 33, W

設定顯示幕的寬度（依照W值）；設定的範圍必須在1與40之間。如果您使用的是 APPLESOFT][，那麼千萬不要將W設定爲0；POKE 33,0 會使 APPLESOFT][完全失效。

如果W值小於33，那麼 PRINT 命令中的第三個停留位置可能將字印出字幕之外。

POKE 34, T

依照T的值設定螢幕的頂端極限，T值的範圍是在0與23之間；0表示頂端的那一個顯示行。 POKE 34,4 將不允許文字資料印在螢光幕的頭

4 行上。請不要將頂端的極限值（ T ）設定爲較底端極限值（ B ）爲小。

POKE 35, B

依照 B 值設定螢幕的底端極限，B 值的範圍是在 0 與 23 之間； 24 就是底端的那一個顯示行。請不要將底端的極限值（ B ）設定爲較頂端的極限值（ T ）爲大。

POKE 36,CH

將游標移到自左邊極限算起的 CH+1 個印字位置上；如果您先設定了左邊的極限爲 6（使用 POKE 32,6 指令），而您想要在螢光幕的第三個位置上印出一個字，那麼您就必須先改變左邊的極限值才可以。CH 值必須小於或等於由 POKE 22,W 指令設定的螢幕寬度，並且必須大於或等於零。這個指令與 HTAB 相似，您可以將游標移到右邊極限的後面，但是它只能使您印出一個字就是了。

POKE 37,CV

將游標移到 CV 值指定的垂直位置上，0 是頂端的顯示行，而 23 則是底端的顯示行。

POKE 50,255　設定白字黑底的狀況。
POKE 50,127　設定閃動的狀況。
POKE 50,63　設定黑白相反的狀況。

POKE 51,42　顯示字＝ "*"
POKE 51,62　顯示字＝ ">"
POKE 51,93　顯示字＝ "]"
POKE 51,33　顯示字＝ "!"

POKE 216,0

將錯誤旗標(ERRFLG)清除，使得正常的錯誤訊息得以發生。

POKE 230,32

打開高解析度圖形的第一頁。

POKE 230,64

打開高解析度圖形的第二頁。

POKE 1011,224

如果您的機器具有自動啓動的 ROM ，那麼使用這個命令可以使您一按 RESET 鍵就重新的 boot 這個系統。

POKE -16368,0 或 POKE 49168,0

將鍵盤輸送綫清除，使得下一個字可以被讀入。這個命令必須立刻跟在讀鍵盤的命令之後。

POKE -16304,0 或 POKE 49232,0

將顯示情況由文字資料轉移到圖形狀況，但是並不把圖形幕清爲黑色。

POKE -16303,0 或 POKE 49233,0

將顯示情況由圖形轉移到文字資料狀況，但是並不重設往上捲螢幕的大

小値。

TEXT 命令使圖形顯示轉移爲文字資料顯示，但是它除了選擇文字資料的第一頁之外，它還將往上捲的值設定爲全螢幕，並把游標放置於螢幕的左下角位置。

POKE -16302,0 或 POKE 49234,0

將圖形與文字資料的混合顯示幕狀況轉移爲全螢幕的圖形狀況。

POKE -16301,0 或 POKE 49235,0

將全螢幕的圖形狀況轉變爲圖形與文字資料的混合狀況，並在螢幕的底端預留 4 行的文字資料幕。

依照其他開關的設定，螢幕的上半部可以用來顯示文字資料、低解析度圖形、或是高解析度圖形。

POKE -16300,0 或 POKE 49236,0

由第二頁轉向第一頁，但是並不把螢幕清除掉，也不將游標移動。當您想要由 APPLESOFT 轉向整數 BASIC 語言的時候，您必須執行這個命令，否則您將仍然看到記憶體的第二頁。

POKE -16299,0 或 POKE 49237,0

由第一頁轉向第二頁，但是並不把螢幕清除掉，也不移動游標。

POKE -16298,0 或 POKE 49238,0

由高解析度圖形頁轉向相對應的文字資料頁，並不把螢光幕清除掉。

POKE -16297,0 或 POKE 49239,0

由文字資料頁轉向相對應的高解析度圖形頁，並不把螢光幕清除掉。

POKE -16296,1 或 POKE 49240,1

將遊戲控制的第 15 個 pin（＃ 0的輸出）設定爲 TTL 高的情況（ 3.5伏）。這是一種關閉的情況。

POKE -16295,0 或 POKE 49241,0

將遊戲控制輸出的第零個接點（＃ 0 ）設定爲 TTL 低的情況（ 0.3 伏）。這是打開這個接點的情況。

POKE -16294,1 或 POKE 49242,0

將遊戲控制輸出的第 1 個接點（第 14 個 pin ）設定爲 TTL 高的情況（ 3.5 伏）。

POKE -16293,0 或 POKE 49243,0

將遊戲控制輸出的第 1 個接點設定爲 TTL 低的狀況（ 0.3 伏）。

POKE -16292,1 或 POKE 49244,1

將遊戲控制輸出的第 2 個接點（第 13 pin ）設定爲 TTL 高的狀況（ 3.5 伏）。

POKE -16291,0　或　POKE 49245,0

將遊戲控制輸出的第 2 個接點設定爲 TTL 低的狀況（ 0.3 伏 ）。

POKE -16290,1　或　POKE 49246,1

將遊戲控制輸出的第 3 個接點（ 第 12 pin ）設定爲 TTL 高的狀況（ 3.5 伏 ）。

POKE -16289,0　或　POKE 49247,0

將遊戲控制輸出的第 3 個接點設定爲 TTL 低的狀況（ 0.3 伏 ）。

PEEK 命令

CH = PEEK(36)

將目前游標在螢光幕上的水平位置讀入 CH 內。這個位置值必定是在 0 與 39 之間，而且是與左邊極限值相關的（由 POKE 32,L 設定的左邊極限值）。因此，如果您使用了 POKE 32,5 設定了左邊的極限值，那麼文字資料幕的最左邊的一個印字位置就是在螢光幕的第 6 個位置上，而 PEEK(36)得到的值就是相關於這個印字位置的值。

CV = PEEK(37)

將目前游標在螢光幕上的垂直位置讀回到 CV 內。 CV 值是絕對位置的值，並不與頂端極限值或是底端極限值發生關連。因此 CV=0 就是在螢幕的頂端， CV=23 則在底端。

```
A = PEEK(69)
```

求得A暫存器內存的值。

```
X = PEEK(70)
```

求得X暫存器內存的值。

```
Y = PEEK(71)
```

求得Y暫存器內存的值。

```
S = PEEK(72)
```

求得S暫存器內存的值。

```
SP = PEEK(73)
```

求得先入後出暫存器指標的值。

```
PE = PEEK (110)
```

求得指向程式結束處指標的值。

```
HM = PEEK (112)
```

求得指向字串表結束處指標的值。

```
If PEEK (216)>127 THEN GOTO 2000
```

如果記憶體位置 216這個數元組的第 7 個數元被設定為 1，那麼就表示一個 ONERR GOTO 指令被執行了。

```
X = PEEK (218) + PEEK (219) * 256
```

如果 ONERR GOTO 指令被執行了，那麼上面的敍述就將 X 設定為發生錯誤情況的敍述的行號。

```
Y = PEEK (222)
```

將 Y 設定為描述錯誤型式的碼（當 ONERR GOTO 的轉移發生的時候）。

```
V = PEEK(-21914)  或  V = PEEK(43622)
```

求得目前磁片的磁片號碼，並把它放入變數 V 內。

```
D = PEEK(-21912)  或  D = PEEK(43624)
```

將目前正在處理的磁碟機號碼放入變數 D 內。

```
S = PEEK(-21910)  或  S = PEEK(43626)
```

將目前正在處理的擴充接點的數目放入變數 S 內。

L = PEEK(-21908) 或 L = PEEK(43628)

將最後被處理的記錄長度放入變數 L 內。

R = PEEK(-21906) 或 R = PEEK(43630)

將最後被處理的記錄數字（ record number ）放入變數 R 內。

B = PEEK(-21904) 或 B = PEEK(43632)

將最後被取得的數元組數目放入變數 B 內。

X = PEEK(-16384) 或 X = PEEK(49152)

讀鍵盤的情況。如果 X>127 就表示鍵盤被按下了一個鍵，而 X 就是這個鍵的 ASCII 值（但是第 7 個數元被設定為 1 ）。這個命令在長的程式中非常有用，您可以利用它做為新資料輸入的檢查工具。

C = PEEK(-16352) 或 C = PEEK(49184)

啓動磁帶輸出（ cassette-output ）一次，使磁帶機發出一聲"嗶"的聲音。

SP = PEEK(-16336) 或 SP = PEEK(49200)

啓動發聲器一次，使發聲器發出"嗶"的聲音。

P0 = PEEK(-16287) 或 P0 = PEEK(49249)

讀入遊戲控制輸出的第 0 個按鈕的情況，如果 P0>127則表示這個按鈕已經被按下了。

P1 = PEEK(-16286) 或 P1 = PEEK(49250)

與 PEEK(-16287) 相同，只是讀取的是第 1 個按鈕的情況。

P2 = PEEK(-16285) 或 P2 = PEEK(49251)

讀取第 2 個按鈕的情況。

CALL 命令

CALL 62454

將高解析度圖形顯示幕依照最接近的 HPLOT 命令所使用的HCOLOR清除乾淨，前面必須有一個繪圖的命令。

CALL 62450

將目前的高解析度螢光幕清爲黑色。

CALL -2458 或 CALL 63078

打開 mini-assembler（只能使用在整數 BASIC 情況下）。

CALL -1998 或 CALL 63538

將整個文字資料的第一頁清爲黑白相反的“ @ ”符號。

CALL -1994 或 CALL 63542

將文字資料第一頁的上端 20 行清爲黑白相反的“ @ ”符號。

CALL -1985 或 CALL 63551

將目前低解析度圖形的顏色值加 3 。

CALL -1728 或 CALL 63808

印出 Y 及 X 暫存器內存的值。

CALL -1724 或 CALL 63812

印出 X 暫存器內存的值。

CALL -1216 或 CALL 64320

設定低解析度圖形狀況。

CALL -1063 或 CALL 64473

使輸出設備發出“ 嗶 ”的一聲。

CALL -1052　或　CALL 64484

使發聲器發出 1/10 秒的“嗶”聲。

CALL -1008　或　CALL 64528

在螢光幕上顯示一個"／"符號並修正游標的位置。

CALL -998　或　CALL 64538

將游標往上移一行。

CALL -958　或　CALL 64578

由目前游標所在的位置到底端極限爲止的文字資料幕清除乾淨。至於在游標之上或是同一行中左邊的字都不受到影響，這個命令與 ESC F 具有相同的功用。

CALL -936　或　CALL 64600

將文字資料幕內所有的字清除乾淨，並把游標移到文字資料幕的左上角位置。這個命令與 ESC @ 及 HOME 命令具有相同的功用。

CALL -926　或　CALL 64610

在螢光幕上造成游標回頭及跳行的功用。

CALL -924 或 CALL 64612

產生一個游標回頭的功用。

CALL -922 或 CALL 64614

產生一個跳行的動作；這與 CTRL J 具有相同的功用。

CALL -912 或 CALL 64624

將螢幕內所有的文字資料往上移動一行。原先最頂端的一行將消失不見，原先的第二行變成頂端的那一行，而底端的那一行成為空白。至於在文字資料幕外的字並不受到影響。

CALL -868 或 CALL 64668

將由游標開始到右邊極限為止的這一行清除乾淨。這個命令與 ESC E 命令相同。

CALL -856 或 CALL 64680

執行一個延遲的迴圈。

CALL -384 或 CALL 65152

在 APPLESOFT][中設定黑白相反的顯示狀況。

CALL -380 或 CALL 65156

在 APPLESOFT][中設定正常的顯示狀況。

CALL -336 或 CALL 65200

跳到 BASIC 系統內，將所有的程式及變數清除乾淨。

CALL -211 或 CALL 65325

印出 ERR 的訊息，並使發聲器發出“嗶”的一聲。

CALL -198 或 CALL 65338

使發聲器發出“嗶”的一聲。

CALL -193 或 CALL 65343

將暫存器的值重新存回暫存器內。

CALL -182 或 CALL 65354

將暫存器內的值儲存起來。

CALL -151 或 CALL 65385

使機器進入監督系統的狀況下。

附錄D

磁碟機作業系統的命令

註記：

我們使用語法來代表一個電腦命令的結構情形。下面是一些簡單的註記，它們用來描述 DOS 命令的語法。

在括弧內，（及），的項目是可以被省略的，這些項目我們一般稱它們爲元素，並不是所有的命令都允許各種元素的出現，但是只要是一個命令可以使用它們，那麼它們就可以依照任何的次序排列（除非有特別的說明）。

如果一個命令中使用了檔的名稱，那麼這個名稱必須緊跟在命令的後面；也就是說，DOS 會把第一個跟在命令後面的項目當做檔的名稱。至於跟在檔名稱後面的其他元素必須使用一個逗號將它們與名字本身分開。

我們在這裏使用“ * ”號代表應該按下那一個鍵：

CTRL 表示當您要打入其他的字時必須先按住 CTRL 鍵。*CTRL*D 就是說您必須按住 CTRL 鍵再打入D字。有時候我們也使用其他的註記表示相同的意思：CTRL-D 與 *CTRL*D 表示同樣的意思。

RETURN 表示按下 RETURN 鍵；至於在每一個命令結束時都必須按下 RETURN 鍵，我們在這裏並不做標示。

RESET 表示按下 RESET 鍵。

ESC 表示按下 ESC 鍵。

大寫的字及逗號表示您必須依照它的格式完整的輸入；小寫的字則表示

您必須自己設定的值。

檔的名稱

檔的名稱可以有 30 個字的長度，而且必須由一個字母開始。名稱內不可以擁有逗號、 CTRL-M 或是 RETURN ，這些都是用來結束命令的。在第一個字母之前的空白將不被理會，而超過 30 個字的部份也不被理會。

當您打入檔的名字時，如果使用了一些像是 Esc 、左箭號、右箭號等的特殊鍵，或是使用了 CTRL 鍵（像是 CTRL-C 、 CTRL-H 等），那麼可能產生意想不到的結果。

如果一個檔的名字內包含了控制字（ control characters ），那麼您將無法看到它們，但是在您想要使用這個檔或消除它的時候，您必須打入它們。

管家的命令 (HOUSE KEEPING COMMAND)

INIT f (,Vv) (,Ss) (,Dd)

範例：INIT HELLO, V18

v 元素定義要被設定格式磁片的磁片號碼。

CATALOG (,Ss) (,Dd)

範例：CATALOG

將指定的或是機器設定的磁碟機內磁片的磁片號碼及檔的名字顯示在螢光幕上。

SAVE f (,Ss) (,Dd) (,Vv)

範例：SAVE COLOR DEMOS, V56

如果磁片上並不存有指定的檔名稱，那麼這個程式檔就會被存到磁片上去了。如果磁片上已經存有這個檔而您所要存入同樣名字但是不同語言或是不同型式的程式，那麼您就會看到 FILE TYPE MISMATCH 的錯誤訊息。

LOAD f (,Ss) (,Dd) (,Vv)

範例：LOAD DOW JONES, V19,D1

在指定的磁碟機內找到具有同樣名字的整數 BASIC 或是 APPLESOFT][程式檔，並把它存入記憶體內。然後您就可以執行它、修改它、或是再將它存入磁片內。 LOAD 命令將所有的文字資料檔關閉，並依照 f 的語言型式自動地修正機器目前使用的語言，同時在把程式檔存入記憶體之前先將記憶體內原先的程式清除掉。

如果您的檔是一個 APPLESOFT][的程式檔，而您的機器內目前並不具有這樣語言，那麼您將得到 LANGUAGE NOT AVAILABLE 的訊息。

LOAD 命令如果不具有任何的元素，那麼它的功用就是由磁帶將一個程式存入記憶體內。

RUN f (,Ss) (,Dd) (,Vv)

範例：RUN ANNUITY, D2

將檔 f 由指定的或是機器設定的磁碟機存入到記憶體內，再執行這個被存入的程式。如果您只是打入 RUN ，那麼只有存在記憶體內的程式會被執行。

RENAME f, g (,Ss) (,Dd) (,Vv)

範例：RENAME SEPERATE, SEPARATE, S4, D1, V0

依照檔名稱 f 到磁碟機內找到相符合的檔，並把它的名字改為 g。檔的內容並不受到改變。如果 f 是一個被打開的檔，那麼在執行這個命令之前它會自動地被關閉。

RENAME命令並不檢查磁片上是否有與g名字相同的檔，所以有可能您使用 RENAME 在磁片上建立了幾個具有相同名稱的檔，這將造成您很大的混亂。

DELETE f (,Ss) (,Dd) (,Vv)

範例：DELETE TEST

將磁片上具有檔名稱f的檔去除；如果f是一個已經被打開的檔，那麼這個命令會將它關閉。

如果f並不存在磁片上，您將看到 FILE NOT FOUND 的訊息，為了避免發生這種情形而使您的程式停止執行，您必須首先 OPEN 這個檔，再DELETE它。

LOCK f (,Ss) (,Dd) (,Vv)

範例：LOCK LOVE LETTERS, V31

這個命令允許您將指定的檔鎖住而不致被改變或消除。一個被鎖住的檔，您在 CATALOG 時可以看到它的前面有一個星號 (*) 。

UNLOCK f (,Ss) (,Dd) (,Vv)

範例：UNLOCK RECIPES, V31,D2

將指定被鎖住的檔打開，這使得您得以修改這個檔內的資料。

VERIFY f (,Ss) (,Dd) (,Vv)

範例：VERIFY SAM

檢查f這個檔內的訊息是否正確地存在磁片上；如果一個檔是正確地存在磁片上的，那麼您將看不到任何訊息，否則您就會得到 I/O ERROR 的訊息。您可以使用 VERIFY 處理任何型式的檔。

MON (C) (,I) (,O)

範例： MON O
　　　MON C, I, O

幾乎在電腦與磁碟機之間傳送的所有命令及訊息都不會顯示在螢光幕上；這個命令使您能夠做到一部份或是全部的顯示工作，這對您的偵錯將有很大的幫助。如果您指出C字，那麼磁碟命令將被顯示；如果您指出I字，那麼由磁碟送往 Apple 的訊息將被顯示出來；如果您用的是O字，Apple 送往磁碟機的訊息也會在螢幕上看到。

當使用這個命令的時候，至少必須指定三個元素中的一個，否則這個命令將不被執行。這些元素排列的時候可以依照任何次序，但是中間必須使用逗號分開。

注意：除非您執行了一個 NOMON 命令，改變語言（ FP 或 INT ）、boot、或是重新啓動系統（ 3D0G ），否則 MON 指令仍然有效。

NOMON (C) (,I) (,O)

範例：NOMON O
　　　NOMON C, I, O

NOMON 命令允許您取消一些或全部的 MON 命令。

三個元素中至少一個必須出現在 NOMON 命令中；這些元素可以依照任何的次序排列，但是中間必須用逗號分開。

MAXFILES n

範例：MAXFILES 6

n是一個1到16的整數，它表示在同一個時間內可以被處理的檔的數目。當執行 MAXFILES 命令時，每一個檔就有595個數元組的記憶體可以

使用。當整個系統 boot 的時候，n 就被設定爲 3，所以您可以順序地打開 3 個檔供您使用。

幾乎所有的 DOS 命令都需要使用一個檔的緩衝區（ file buffer ）；所以如果您設定 MAXFILES 1 的命令，而又把一個檔打開了，那麼在您要執行一個 DOS 命令時（像是 CATALOG 命令），您就會看到 NO BUFFERS AVAILABLE 的訊息。

使用 MAXFILES 命令會使 HIMEM 被移動，這樣往往會把整數 BASIC 程式或 APPLESOFT][的字串消除掉；因此，您在執行程式之前最好先設定好 MAXFILES 的值。

處理的命令

FP (,Ss) (,Dd) (,Vv)

範例：Fp,D2

這個命令使您的系統進入 APPLESOFT][BASIC 。任何存在記憶體內的整數 BASIC 程式或是 APPLESOFT][程式都將被清除。

INT

範例：INT

這個命令使 Apple 進入整數 BASIC 狀況下，任何存在記憶體內的整數 BASIC 或是 APPLESOFT][程式都將被清除。

CTRL-D （ 也可寫爲 *CTRL*D)

範例：

```
10 D$=CHR$(4)
20 PRINT D$;"WRITE CHESS"
```

每一個由 Apple 來輸出的字都必須經過 DOS 的檢查；如果 Apple 印出一個 RETURN 字（大多數的 PRINT 敍述自動地都是以一

個 RETURN 做爲結束的），而下一個字是 CTRL-D ，那麼 DOS 就知道隨後的字（到下一個 RETURN 爲止）就是一個 DOS 命令了。大多數的 DOS 命令都可以在一個整數 BASIC 或是 APPLESOFT][的程式中使用，使用的方法就是在一個 PRINT 敍述中用一個由 CTRL-D 開頭而後面跟著 DOS 命令的字串。

PR# s

範例：PR# 6

將隨後的輸出送到由# s的擴充接點控制的外圍設備去。 PR# 0 使輸出的資料被送到螢光幕上。如果您將這個命令用在程式內，它必須用 PRINT DOS 的命令表示就像下面一般；

```
10 D$="":REM CTRL-D
20 PRINT D$;"PR# 1"
```

如果擴充接點上沒有任何的控制卡存在，那麼您就必須按下 RESET 鍵，才能使機器回復正常的狀況。

IN# s

範例：IN# 6

由接在# s擴充接點的外圍設備接收輸入的資料。 IN# 0 使機器由鍵盤接受資料的輸入。如果在程式中使用這個命令，它必須用 PRINT DOS 的命令表示，如下面一般：

```
10 D$="":REM CTRL-D
20 PRINT D$;"IN# 1"
```

如果# s的擴充接點上沒有任何的控制卡，那麼您就必須按下 RESET 鍵使機器回復正常。

CHAIN f (,Ss) (,Dd) (,Vv)

範例：CHAIN PART TWO, D1,S7,V0

這個命令只能使用在整數 BASIC 中，它將一個整數 BASIC 的程式存入記憶體內並執行它，但是並不把任何的變數清爲零。這就是說程式 f 可以利用前面執行過的程式所產生的結果做爲本身的資料；並且也可以保留住本身執行的結果供以後的程式使用。在 APPLESOFT][中，您不可以使用這個命令。

順序處理檔的命令
(SEQUENTIAL TEXT FILE COMMANDS)

OPEN f (,Ss) (,Dd) (,Vv)

範例： OPEN SESAME, D2

預先在記憶體內預留 595 個數元組的記憶體位置供文字資料檔 f 使用，並且準備由這個檔的開始處讀出或寫入資料。這個命令與 WRITE 及 READ 命令合用來建立或取得一個順序處理檔內的資料。

如果在指定的磁碟片上檔 f 並不存在，那麼這個命令就建立一個新的檔。如果檔 f 已經被打開了，那麼它將先被關閉再被打開。

CLOSE (f)

範例：CLOSE WINDOW

如果您正在寫入（ WRITE ）資料的時候使用了這個命令，那麼它會使存在檔暫存區（ file buffer ）內輸出部份的所有字被寫入磁片內。如果

您只是使用 CLOSE 命令而沒有指定任何檔的名字，那麼所有被打開的檔都將被關閉，只有 EXEC 檔是一個例外（在任何時間內只有一個 EXEC 檔可以被打開，如果有另外一個 EXEC 檔在隱藏的情況下被打開了，那麼目前存在的這個 EXEC 檔立刻就被關閉）。

當一個文字資料檔被執行中的程式打開了，而這個程式又被 CTRL-C 所打斷，那麼您最好再執行 CLOSE 命令，以免失掉存在記憶體內的資料。

被 OPEN 命令打開的檔一定要在結束處理過程之後，使用 CLOSE 命令將它關閉，以免失掉某些資料。

WRITE f (,Bb)

範例： WRITE ADDRESS.DATA

在這個指令之後的所有 PRINT 敍述會把它的輸出送到指定的檔內而不是送到螢光幕上。使用B元素時，寫入的動作由第b個數元組開始。

這個寫入的指令將被任何的 DOS 命令或是 INPUT 命令取消，即使是虛的 DOS 命令（只是在 PRINT 敍述中使用CTRL-D）也能夠達到這個取消的效果。 WRITE 命令必須在程式執行的狀況下使用。

在這個命令之後所有 Apple 的輸出訊息都將被寫入磁片上，這些訊息包括了 INPUT 敍述中的問號、錯誤訊息、以及其他一些您並不想要的字。

READ f (,Bb)

範例： READ SESAME

在這個命令之後，所有的 INPUT 敍述（以及 APPLESOFT][中的 GET 敍述）都會由指定的順序處理檔內去接收資料，而不是由鍵盤上去得到資料，如果您使用了B元素，那麼讀的動作就由第b個數元組開始。

INPUT 敍述使得存在順序處理檔內的資料以區（ field ）的方式讀入——每一次一個區，一個區可以由1到32767個字所組成，並且以一個 RETURN 做爲結束。但是由於字串長度的限制及緩衝區的儲存能力有限，因

此對於超過 255 個字長度的區都是較難處理的。

READ命令同樣的可以被任何在 PRINT 敍述中的 DOS 命令所取消（只是一個 PRINT CTRL-D 就可以達到這個目的）。 READ 命令必須被使用在程式執行的情況下。

APPEND f (,Ss) (,Dd) (,Vv)

範例：APPEND MORE INFO

這個命令將順序處理的文字資料檔打開，但是將指到檔內位置的指標指到檔結束的地方；在這個命令之後寫入檔內的字將被接著放在原先檔內最後一個字的後面。 APPEND 命令之後必須緊跟著一個 WRITE 命令（具有同樣檔的名稱）。 APPEND 命令之後絕對不可以跟著 OPEN 命令，因爲 OPEN 命令會把指標指到檔開頭的位置。

POSITION f (,Rp)

範例：POSITION ADDRESS.DATA, R277

POSITION命令將指標指到檔內您所指定的第 P 個區的開始位置（一個區就是一連串的字後面跟著一個 RETURN ）。隨後的 READ 或 WRITE 都會由這個位置開始處理檔內的資料。

POSITION 命令是一個處理相關位置的命令而不是一個絕對位置的命令，因爲當您使用它時，它是依照您目前在檔內的位置而往後計算的。

POSITION 命令實際上在檔內是一個字一個字的掃瞄直到找到第 P 個 RETURN 字爲止；如果它一直找不到第 P 個 RETURN ，那麼它就會把指標指到尚未被存入資料的第一個數元組，然後您就會看到 END OF DATA 的訊息了；這種情形一般都是當第 P 個區域是位在目前指標所指的位置之前，或是這個區域是在檔內最後一個資料的後面的時候發生的。

EXEC f(,Rp) (,Ss) (,Dd) (,Vv)

範例：EXEC UTILITY

與 RUN 命令相似，只是檔 f 是一個文字資料檔，而且裏面儲存的是 BASIC 及 DOS 的命令。這種方式使得您可以設定一個檔來控制 Apple ，就如同您自己控制它一般。

在同一個時間內只能有一個 EXEC 檔被打開，如果一個 EXEC 檔內存有立即執行的 EXEC 命令，那麼原先的這個檔就被關閉而這個 EXEC 檔則被打開並且執行。如果 EXEC 命令打開了一個檔， CLOSE 命令並不能將 EXEC 檔關閉； EXEC 檔在執行完畢了之後，它會自動地把它自己關閉並停止執行。如果一個被 EXEC 命令執行的檔內存著有執行其他程式的命令，那麼 EXEC 命令將會等待這個程式執行完畢，然後才繼續地執行 EXEC 檔內的下一個命令。

然而，當一個程式正在被執行而一個 EXEC 檔也被打開的時候，程式中的 INPUT 敘述會將 EXEC 檔內下一個區的資料當做輸入，而不理會鍵盤。更壞的是，如果這一個區是一個立即執行的 DOS 命令，那麼這個命令會在程式繼續執行之前先被執行。

如果在 EXEC 檔被打開的時候，您使用 CTRL-C 打斷一個 APPLE SOFT][程式的執行，那麼存在 EXEC 檔內剩下的命令將不會再被執行了。

如果您在 EXEC 命令中使用了R元素將指標指到檔內的第P個區域的開始處，那麼 EXEC 命令將由這一點開始執行。

如同在 POSITION 命令中一般，R元素是一個相關位置的元素；但是 EXEC 與 POSITION 不同的一點是 EXEC 永遠由檔的起頭處開始算起，因此P永遠是與0相關的。至於其他的元素與以前的用法相同。

如果您指定的R元素超過了檔的長度，那麼您將得到 END OF DATA 的訊息。

隨機處理檔的命令

OPEN f, Lj (,Ss) (,Dd) (,Vv)

範例：OPEN SESAME, L2

OPEN 命令爲檔 f 準備好了 595 個數元組的檔緩衝區供這個隨機處理的文字資料檔使用，並且依照 j 的值設定記錄的長度。j 值必須在 1 與 32767 之間，而它的機器設定值爲 1 。

OPEN 命令與 READ 及 WRITE 命令合用來處理一個隨機處理的文字資料檔。請注意的是，L 元素必須在命令中指出，也就是說您必須設定一個隨機處理檔內記錄的長度；每一次當您要想使用某一個隨機處理檔時，您必須使用 OPEN 命令與同樣的 L 元素值打開這個檔，這樣子 DOS 才能夠使用這個值計算出您所要使用的記錄的位置。如果您在使用這個命令的時候，檔 f 並不存在磁片上，那麼這個命令就建立一個新的檔。

CLOSE (f)

範例：CLOSE BOOK

如果您正在將資料寫入檔內，那麼一個 CLOSE 命令就會使存在檔緩衝區內輸出部份的剩餘資料被送入磁片內。 CLOSE 命令使原先在 OPEN 命令中供檔 f 使用的緩衝區不再存在。如果在一個 CLOSE 命令中並不包含任何檔的名稱，那麼所有被打開的檔都將被關閉（除了 EXEC 檔之外）。

如果在一個文字資料檔被打開的時候，而執行中的程式被 CTRL-C 所打斷，那麼您最好打入 CLOSE 命令以免失掉資料。

所有被 OPEN 命令打開的檔都必須使用 CLOSE 命令將它們關閉起來，否則您可能會失掉存在記憶體內的資料。

WRITE f (,Rr) (,Bb)

範例：WRITE ADDRESS.DATA, R3

在這個命令之後的 PRINT 敍述會把輸出的字送到指定的檔內，而不是送到螢光幕上。 WRITE 命令可以被加在 PRINT 敍述內的任何DOS命令及 INPUT 命令取消； WRITE 只能出現在程式執行的 PRINT 敍述內。

R（記錄）元素使得寫入的動作由第 r 個記錄的第一個數元組開始，而每一個記錄內所擁有的數元組數目都是依照 OPEN 命令中的L元素而設定的。r的機器設定值是0；如果您使用了B元素在這個命令中，那麼寫入的動作就會由第 r 個記錄的 b 個數元組開始。

在 WRITE 命令之後所有 Apple 的輸出字都會被送到磁片上去，這些字包括了 INPUT 敍述產生的問號、錯誤訊息、以及其他您不需要的字。使用一個虛的 DOS 命令取消它的作用是最簡易的方法。

READ f (,Rr) (,Bb)

範例：READ SESAMES,R3,B30

在這個敍述之後的 INPUT 敍述（以及 APPLESOFT][中的GET敍述）都會由您指定的檔內取得資料，而不是如往常一般的由鍵盤得到資料。 INPUT 敍述使得資料由指定的隨機處理檔內目前被指到的記錄內被讀出（一次讀出一個區）。

一個區可以由 1 到 32767 個字所組成，並且以 RETURN 字作為結束。然而，任何記錄內所包含的字數都不能超過在 OPEN 命令中所指定的 j 值。

R（記錄）元素使讀出的字由第 r 個記錄的第一個數元組開始，r 的設定值是0；如果您使用了B元素，那麼讀出的動作就由第 r 個記錄的第 b 個數元組開始了。

READ命令可以由包含在 print 敍述內的 DOS 命令所取消，您取消它最好的方法是用一個虛的 DOS 命令。

機器語言檔的命令

```
BSAVE f, Aa, Lj (,Ss) (,Dd) (,Vv)
```

範例：
```
BSAVE PICTURE, A16384, L8192
BSAVE PICTURE, A$4000, L$200
```

這個命令建立一個名稱為 f 的檔，並且將記憶體內一段區域內的資料存入這個檔內。這一段記憶體的起始位置由 a 指定，而整段的長度則由 j 決定。

上面的兩個例子就是由高解析度圖形的第二頁將一個高解析度圖形存入磁片內。第二個例子使用的是十六進位的方式表示數字，而第一個例子則是用十進位的方式表示。

```
BLOAD f (,Aa) (,Ss) (,Dd) (,Vv)
```

範例：
```
BLOAD PICTURE, A8192
BLOAD PICTURE, A$2000
```

如果 a 元素沒有被指定，那麼 BLOAD 命令就依照原先 BSAVE 時所指定的位置把這個檔放到同樣的記憶體內；如果指定了 a 元素，這個檔則被放入指定位置的記憶體內；但是請注意一點，那就是一個機器語言的程式經過這樣不同位置的搬動之後，可能就無法再被執行了。

假設現在已經有一個高解析度圖形被 BSAVE 命令存在磁片上，它的名字就是 PICTURE ；那麼在第一個例子中的命令就把這個檔存放到記憶體內高解析度圖形的第一頁內。第二個例子與第一個例子具有相同的效果，只是它用來表示位置的數字是十六進位系統的數字罷了。

```
BRUN f (,Aa) (,Ss) (,Dd) (,Vv)
```

範例：
```
BRUN  SUPER, A$C0A, v75
```

這個命令首先執行一個 BLOAD 命令將檔 f 存入由 a 開始的記憶體內；如果您省略了 A 元素，那麼這個檔就是由與 BSAVE 命令中使用的同樣

的位置開始存入記憶體。等到 BLOAD 命令執行完畢之後，這個機器語言的程式檔就由一個 JMP 命令跳到 a 位置開始執行了。

附錄E

錯誤的訊息

APPLESOFT II 的錯誤訊息

碼	錯誤訊息
0	NEXT without FOR
16	Syntax error
22	RETURN without GOSUB
42	Out of DATA
53	Illegal Quantity
69	Overflow
77	Out of Memory
90	Undefined Statement
107	Bad Subscript
120	Redimensioned Array
130	Division by Zero
163	Type Mismatch
176	String Too Long
191	Formula Too Complex
224	Undefined Function
254	Bad Response to INPUT Statement
255	CTRL-C Interrupt Attempted

DOS的錯誤訊息

碼	錯誤訊息
1	LANGUAGE NOT AVAILABLE

2,3	RANGE ERROR
4	WRITE PROTECTED
5	END OF DATA
6	FILE NOT FOUND
7	VOLUME MISMATCH
8	I/O ERROR
9	DISK FULL
10	FILE LOCKED
11	SYNTAX ERROR
12	NO BUFFERS AVAILABLE
13	FILE TYPE MISMATCH
14	PROGRAM TOO LARGE
15	NOT DIRECT COMMAND

整數BASIC的錯誤訊息

第一組的訊息

這些訊息一般都發生在您打入程式或一個指令的時候。

*** SYNTAX ERR

這是一個最常見的錯誤訊息，它的意思就是說“我不瞭解您打入的東西”。這時候您就必須重新修正您打入的指令了。像這種情形的錯誤是最容易修正的了。

*** TOO LONG ERR

這個訊息表示您打入了超過127個字長度的一個敍述。當您在一行中打入了超過255個字長度的訊息時，您將會看到一個"／"符號，雖然螢光幕上仍然看的到這些訊息，但是電腦已經將它們忘記了。

*** >32767 ERR

您打入了一個較32767爲大或較－32767爲小的數字。

*** >255 ERR

在 BASIC 中往往有許多的值不能小於零或大於255；例如在一個 PLOT 敍述中的值就是這樣。當您這麼做了的時候，上面這個錯誤訊息就是您見到的。

第二組的錯誤訊息

這一類的訊息一般都發生在程式執行的時候。顯現這類訊息的同時，您也可以看到像是 STOPPED AT 390 之類的訊息，它們是用來告訴您這個錯誤發生的敍述的行號的。

*** BAD BRANCH ERR

當您想要使用一個 GOTO 或是 GOSUB 指令轉移到一個並不存在的行號上去時，您就會看到這個訊息。

*** BAD RETURN ERR

您必須在一個由 GOSUB 執行的副程式中使用 RETURN 命令；上面這個訊息就發生在當一個 RETURN 命令並沒有被使用在副程式中的時候。或者說，您想要執行較 GOSUB 數目爲多的 RETURN 命令，您就會看到這個訊息。

*** BAD NEXT ERR

在執行 NEXT 之前，必須先有FOR敍述被執行才可以。當您想要執行一個前面並沒有FOR敍述的 NEXT 指令時，您就會得到這個訊息。FOR及 NEXT 命令必須有相同的變數名稱；例如 FOR X=2 TO 5 及 NEXT X 。

*** 16 GOSUBS ERR

這個訊息表示在一個副程式中使用了一個 GOSUB 命令到另一個副程式中執行，而這個副程式中又有一個 GOSUB 命令轉移到另一個副程式，如此繼續下去共有 16 個層次，您就會得到這個訊息。像下面這個程式：

```
100 GOSUB 200
200 GOTO  100
```

會使 GOSUB 一直被執行而沒有任何的 RETURN ，那麼您就會得到這個訊息。在一個程式中您可以有任何數目的副程式，像上面的限制只發生在副程式聯結（ nested ）的情況下。

*** 16 FORS ERR

這個訊息表示有 16 層聯結的 FOR 迴圈。

*** NO END ERR

這個訊息表示您在程式的結尾沒有加上 END 敍述。雖然您看到這個訊息，但是它對整個程式都沒有任何的影響。

*** MEM FULL ERR

這個訊息只發生在您設計的程式或程式中需要使用的資料使您的 APPLE][機器的記憶體無法儲存的時候。這個訊息也可能發生在由磁帶存入程式的過程中，但這時候並不一定表示記憶體已經滿了。

*** TOO LONG ERR

如果您對 INPUT 的反應輸入了較128個字爲多的資料，您就會得到這個訊息。這種訊息在包含了有 12 層聯結括弧以上的運算式子中也會發生。

*** DIM ERR

如果一個變數已經被設定了行列值，而您又想重新設定它的行列值，那麼您就會得到這個訊息。

*** RANGE ERR

這種錯誤訊息可以發生在下面幾種情況：矩陣內的標註值超過了 DIM 敍述中所定義的值，或者標註值較 1 爲小；如果在 HLIN 、 VLIN 、 PLOT 、 TAB 或 VTAB 中的元素值超過了範圍，您也會看到這個訊息。

*** >32767 ERR

如果在一個 INPUT 敍述中接收了或是在一個計算式子中得到了一個大於 32767 或是小於 －32767 的值，您就會看到這個訊息。

*** >255 ERR

在 BASIC 中的某些元素值必須大於零或小於 255；例如在 COLOR

PLOT 及 TAB 敍述中的元素值就是這樣。如果您的程式以及在 INPUT 敍述的回應中得到了超過這個範圍的值，您就會看到這個訊息。

*** STR OVL ERR

您如果在一個字串中放入較 DIM 敍述所設定的值爲多的字，那麼您就會看到這個訊息了。

*** STRING ERR

幾乎在字串中發生的任何錯誤都會使這個訊息產生。

STOPPED AT XXX

XXX是一個行號；這個訊息往往跟著一個錯誤訊息，使您能夠知道這個錯誤是在那一個敍述發生的。當您使用 CTRL-C 打斷程式的執行時，您也會看到這個訊息的。

#

下面有兩種不同的訊息都是由#號（ # ）開頭的。第一類的例子就是：

#120 #130 #250

等等繼續下去；這些都是由於執行 TRACE 命令產生的結果。您可以使用 NOTRACE 命令取消這個功用。

另外一種就像：

#15 I=55 #20 J=3

等等繼續下去；這是 DSP 指令執行的結果，它使在 DSP 命令中所有的變數及它的值與行號同時顯現。如果您不想要這樣的輸出，那麼您只要把DSP敍述消去就可以了。

第三組的錯誤訊息

這些訊息只有發生在對 INPUT 敍述做反應的時候。

*** TOO LONG ERR

您打入了超過128個字的資料時，您就會看到這個訊息。

RETYPE LINE

注意這個訊息的前面沒有三個星號。它表示您打入的資料並不是這個 INPUT 敍述所期望的資料型式，您必須重新打入；程式並不會因爲這個訊息而停止執行。這種情形一般都發生在輸入資料的型式錯誤，或是輸入不正確數目的數字的時候。

第四組的錯誤訊息

這些錯誤發生在由磁帶將程式存入記憶體內的時候。

```
ERR
*** MEM FULL ERR
ERR*** MEM FULL ERR
```

上面三種錯誤都表示同一件事：那就是資料無法由磁帶正確的讀入記憶體內。當存入記憶體的時候， *** MEM FULL ERR 一般表示磁帶內的資料無法正確的讀入；但是如果您的機器過小（ 4 K ），那麼它也可能表示程式太長的意思。

如果在磁帶上的程式並不是太長，那麼就表示音量不是太高就是太低，而音調的控制也沒有開到最大，或者說是磁帶機本身的一些問題。對於檢查磁帶機的步驟，您必須參考 Apple][的手册。

附錄F

保留字

APPLESOFT Ⅱ 的保留字

175 &			
212 ABS	205 AND	230 ASC	197 AT
225 ATN	140 CALL	231 CHR$	189 CLEAR
160 COLOR=	187 CONT	222 COS	131 DATA
184 DEF	133 DEL	134 DIM	148 DRAW
128 END	221 EXP	159 FLASH	194 FN
129 FOR	214 FRE	190 GET	176 GOSUB
171 GOTO	136 GR	146 HCOLOR=	145 HGR
144 HGR2	163 HIMEM:	142 HLIN	151 HOME
147 HPLOT	150 HTAB	173 IF	139 IN#
132 INPUT	211 INT	158 INVERSE	232 LEFT$
227 LEN	170 LET	188 LIST	182 LOAD
220 LOG	164 LOMEM:	234 MID$	191 NEW
130 NEXT	157 NORMAL	198 NOT	156 NOTRACE
180 ON	165 ONERR	206 OR	216 PDL
226 PEEK	141 PLOT	185 POKE	161 POP
217 POS	186 PRINT	138 PR#	135 READ
167 RECALL	178 REM	174 RESTORE	177 RETURN
233 RIGHT$	219 RND	152 ROT=	172 RUN
183 SAVE	153 SCALE=	215 SCRN(	210 SGN
154 SHLOAD	223 SIN	195 SPC(	169 SPEED=
218 SQR	199 STEP	179 STOP	168 STORE
228 STR$	192 TAB(	224 TAN	137 TEXT
196 THEN	193 TO	155 TRACE	213 USR
229 VAL	143 VLIN	162 VTAB	181 WAIT
XPLOT	149 XDRAW		

注意： XPLOT 是一個保留字而不是一個命令。

整數BASIC的保留字

ABS	AND	ASC	AT
AUTO	CALL	COLOR=	CON
DEL	DIM	DSP	END
FOR	GOSUB	GOTO	GR
HIMEM	HLIN	IF	IN#
INPUT	LEN	LET	LIST
LOAD	LOMEM	MAN	MOD
NEW	NEXT	NOT	NOTRACE
OR	PDL	PEEK	PLOT
POKE	POP	PRINT	PR#
REM	RETURN	RND	RUN
SAVE	SCRN	SGN	STEP
TAB	TEXT	THEN	TO
TRACE	VLIN	VTAB	

DOS的保留命令

APPEND	BLOAD	BRUN	BSAVE
CHAIN	CLOSE	DELETE	EXEC
INIT	LOAD	LOCK	OPEN
POSITION	READ	RENAME	RUN
SAVE	UNLOCK	VERIFY	WRITE

附錄G

編輯命令

我們在 Apple][上使用編輯命令(editing commands)使游標自由地移動。在使用這些命令之前必須先按下 ESC 鍵進入編輯狀況。

在進入了編輯狀況之後，我們就可以使用鍵盤上 I， J， K，M 四個鍵移動游標了。M 及 I 用來使游標往上及下移動；K 及 J 則使游標往左及右移動。這種情形可以由圖A－1中瞭解。當想要離開編輯狀況的時候，您只需按下空白鍵就可以了。

```
        +---+
        ! I !
        +---+
    +---+ +---+
    ! J ! ! K !
    +---+ +---+
       +---+
       ! M !
       +---+
```

圖A－1　編輯鍵

由一行中清除一個或更多的字

首先進入編輯狀況，然後將游標移到想要被移開字的第一個位置。這時候按下空白鍵離開編輯狀況，再連續地按空白鍵直到所有您不需要的字都被去除了爲止。

將到一行結尾爲止的字全部清除掉

進入編輯狀況之後，將游標移到您想要去除行的第一個字，然後按下E鍵就可以把這一行到結尾爲止的字都去除了。

將到螢幕底端爲止的所有字都清除掉

在編輯狀況下，將游標移到您想要去除區域的第一個字，然後按下F鍵就可以達成目的了。

將整個螢幕清除

在編輯狀況下，按下@鍵就可以將整個螢幕清除乾淨了。這個時候游標會在螢幕的左上角閃動。

I, M, J, K 四個鍵只能對螢光幕上的字產生影響，如果您想要修改存在記憶體中的內容，那麼您必須使用左箭號及右箭號，它們分別代表清除及複製的意思。請參看圖A－2。

```
+---+  +---+
! < !  ! > !
+---+  +---+
```

圖A-2　清除及複製鍵

按下消除鍵您就可以將游標往回移動一個位置，同時也把最後輸入的字消除了。如果最後的一個字是一個控制字，那麼 Apple][就視它爲沒有輸入的情況，但是在螢光幕上則並沒有任何的改變。

按下複製鍵可以使您將游標往前移動一個位置，同時把它經過的字複製到記憶體內。

這兩個鍵可以與 REPT 鍵合用以便快速地達到消除及複製的目的。

附錄H

設定的函數

SECANT:
SEC(X) = 1/COS(X)

COSECANT:
CSC(X) = 1/SIN(X)

COTANGENT:
COT(X) = 1/TAN(X)

INVERSE SINE:
ARCSIN(X) = ATN(X/SQR(-X*X+1))

INVERSE COSINE:
ARCCOS(X) = -ATN(X/SQR(-X*X+1))+1.5708

INVERSE SECANT:
ARCSEC(X) = ATN(SQR(X*X-1))+(SGN(X)-1*1.5708

INVERSE COSECANT:
ARCCSC(X) = ATN(1/SQR(X*X-1))+(SGN(X)-1)*1.5708

INVERSE COTANGENT:
ARCCOT(X) = -ATN(X)+1.5708

HYBERBOLIC SINE:
SINH(X) = (EXP(X)-EXP(-X))/2

INVERSE HYPERBOLIC COSINE:
ARCCOSH(X) = LOG(X+SQR(X*X-1))

INVERSE HYPERBOLIC COTANGENT:
ARCCOTH(X) = LOG((X+1)/(X-1)/2

INVERSE HYPERBOLIC COSECANT:
ARCCSCH(X) = LOG((SGN(X)*SQR(X*X+1)+1)/X)

INVERSE HYPERBOLIC SECANT:
ARCSECH(X) = LOG((SQR(-X*X+1)+1/X)

INVERSE HYPERBOLIC SINE:
ARCSINH(X) = LOG(X+SQR(X*X+1))

INVERSE HYPERBOLIC TANGENT:
ARCTANH(X) = LOG((1+X)/(1-x))/2

HYPERBOLIC COTANGENT:
COTH(X) = EXP(-x)/(EXP(X)-EXP(-X))*2+1

HYPERBOLIC COSECANT:
CSCH(X) = 2/(EXP(X)-EXP(-x))

BASE A LOGARITHM OF X
LOGa(X) = LOG(X)/LOG(a)

BASE 10 LOGARITHM OF X
LOG10(X) = LOG(X)/2.30258509

MODULO a
MODa(X) = INT((INT((X/a-INT((X/a))*a+.05)*SGN(X/a)

HPERBOLIC SECANT:
SECH(X) = 2/(EXP(X)+EXP(-X))

HYBERBOLIC TANGENT:
TANH(X) = -EXP(-X)/EXP(X)+EXP(-X))*2+1

PUBLISHED & PRINTED IN HONG KONG H.K.$ 35.00

APPLE BASIC 應用設計

張其邦 譯

博文出版社出版